DOCUMENT RESUME

ED 176 991 SE 029 030

AUTHOR	Luehrmann, Arthur W., Ed.
TITLE	Proceedings of the Second Annual Conference on Computers in the Undergraduate Curricula (Dartmouth College, Hanover, New Hampshire, June 23-25, 1971).
INSTITUTION	Dartmouth Coll., Hanover, N.H.
SPONS AGENCY	National Science Foundation, Washington, D.C.
PUB DATE	71
NOTE	562p.; Contains occasional marginal legibility in figures and computer printouts
EDRS PRICE	MF02/PC23 Plus Postage.
DESCRIPTORS	Biology; Calculus; Chemistry; *Computer Oriented Programs; *Computers; *Conference Reports; Economics; *Higher Education; Humanities; Instructional Media; Mathematics Education; Physics; Probability; Social Sciences; *Undergraduate Study
IDENTIFIERS	Dartmouth College NH

ABSTRACT

Presented are 70 papers given at the Conference on Computers in the Undergraduate Curricula as well as a listing of over 100 papers submitted but not presented. The papers presented describe a wide variety of ideas and techniques for improving the teaching of traditional subjects through adjunctive use of the computer. (MK)

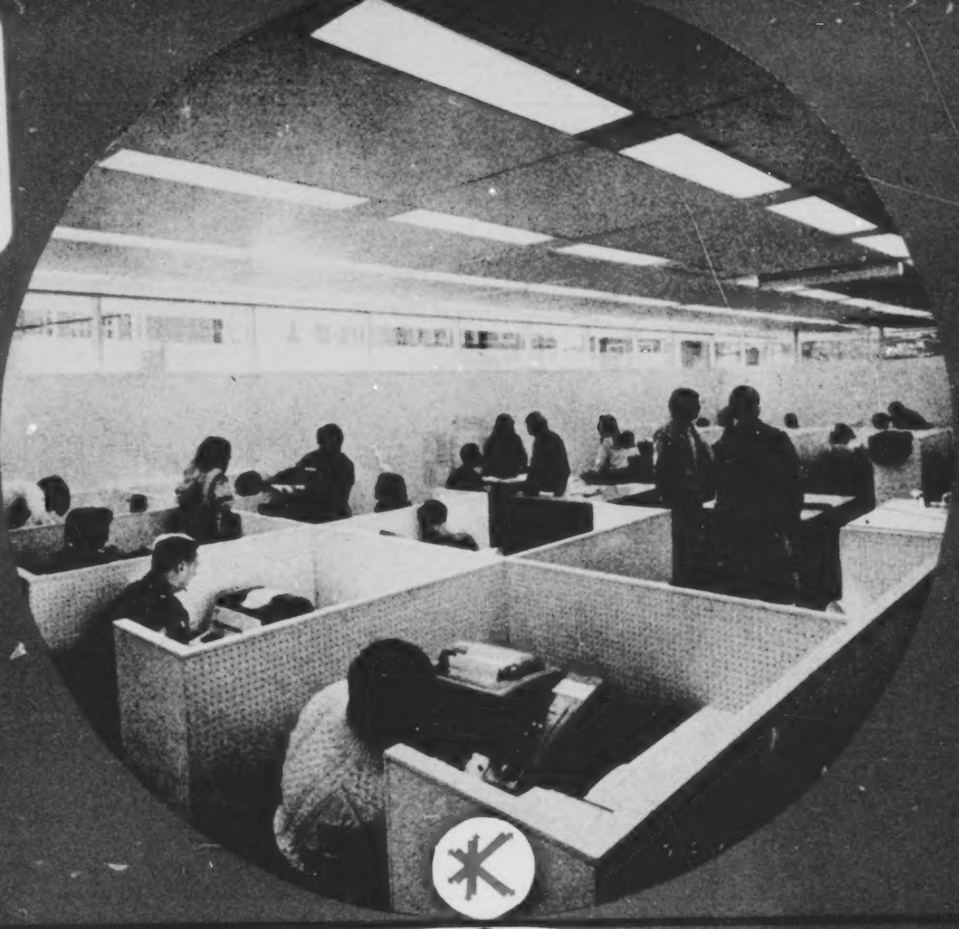
****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

U S DEPARTMENT OF HEALTH.
EDUCATION & WELFARE
NATIONAL INSTITUTE OF
EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
Mary L. Charles
NSF
TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

ED176991



Conference on Computers in the Undergraduate Curricula

June 23, 24, 25
1971
Dartmouth
College

SE 029 030

PROCEEDINGS
OF THE
SECOND ANNUAL

Conference on Computers in the Undergraduate Curricula

June
23, 24, 25, 1971
Dartmouth
College
Hanover,
New Hampshire

SPONSORED BY
DARTMOUTH COLLEGE
WITH SUPPORT FROM
THE NATIONAL
SCIENCE
FOUNDATION

EDITED AND PUBLISHED BY

Computer Oriented Materials Production for Undergraduate Teaching

Copyright © 1971 by the Trustees of Dartmouth College, Hanover, New Hampshire

Library of Congress Catalog Card number 79-165540 Printed in the United States of America

Distributed by The University Press of New England, Box 979, Hanover, New Hampshire 03755

FOREWORD

The importance of computing to the educational process has long been recognized. Even prior to the release of the "Pierce Report" in 1967, many teachers at numerous institutions of higher education had committed themselves to improved teaching through appropriate use of computers. But progress has been slow and hard, partly because there existed no structure or professional society to foster the exchange of ideas and to advance the level of expertise. Sensing the need of such teachers to share ideas and benefit from the work of their colleagues at other institutions, as well as the need for professional recognition, the National Science Foundation and the University of Iowa supported the first Conference on Computers in the Undergraduate Curricula. About 750 persons attended and participated. The great success of this Conference encouraged Dartmouth College to sponsor a similar Conference, held on June 23-25, 1971, again with National Science Foundation support.

About 70 papers (of 285 submitted) were accepted by the reviewing panels for presentation. These papers describe a wide variety of ideas and techniques for improving the teaching of traditional subjects through adjunctive use of the computer. In keeping with this narrow view of the purpose of the Conference, the reviewing panels excluded from consideration papers in areas such as computer science, programming, and educational research.

As is the custom with some research journals, a number of papers are listed by title. The authors of somewhat more than half of the 215 papers not presented indicated their willingness to have their work so listed. Hopefully, this listing will further help teachers to identify colleagues at other institutions doing similar work.

All of the papers scheduled for presentation at the Conference are included in these Proceedings. For the two panel discussions, abstracts or position statements are included, in lieu of complete papers. The text of the keynote address by Dartmouth President John G. Kemeny is also not included.

Thomas E. Kurtz

Thomas E. Kurtz
Chairman, Organizing Committee

ACKNOWLEDGMENTS

Presenting a large national conference is like producing a Wagner opera. It takes ten people in the pit for every person on stage. I should like now to turn the spotlight on a number of the off-stage principals in this year's summer production.

If any one person deserves the title of impresario it must be Arthur S. Melmed, who for so many years has, on the one hand, nurtured and encouraged our work in this burgeoning field and, on the other, been a stern critic and a guardian of the concomitant investments of public funds. He, and more recently Andrew R. Molnar and others on the staff of the Office of Computing Activities of the National Science Foundation, have with Stanislavskian insistence caused us to examine our roles and our motives. The result of this introspection is a national community of self-aware practitioners and devotees, some 700 of whom gathered in Iowa City in June of last year. Like that gathering, this year's meeting is underwritten in part by NSF/OCA funds.

Credit for general planning of the 1971 meeting belongs to the Conference Organizing Committee. Thanks are due to Fred W. Weingarten for coordinating the reviewing effort, to John W. Hamblen for handling conference publicity, to Gerard P. Weeg for passing along mailing lists and lore from the Iowa Conference, and to Glenn R. Ingram for advice and counsel. To the Dartmouth members of the committee, Thomas E. Kurtz and myself, fell the responsibility for general coordination and Proceedings publication, respectively. It was Tom who orchestrated and, in his quietly persuasive style, directed the Conference at Dartmouth. On him rested the burden of most major policy decisions as well as a host of trivial but essential matters. Many members of the staff of the Kiewit Computation Center, led by Thomas E. Byrne and Eugene A. Fucci, formed an occasionally high-pitched chorus of assistants. (There were no supernumeraries in this production.)

When Ted Sjoerdsma, who was charged with publishing the Proceedings of the Iowa Conference last year, heard that I had the job this year, he gave out a Mephistopholean laugh. As time went on it became clear what he meant. Moving from seventy raw manuscripts through author proofs, through editorial changes, and on to 530 pages of camera-ready copy in eight weeks was a herculean effort. Hercules, in this case, was played by my associate in Project COMPUTe, A. Kent Morton. Kent flow-charted and tack-boarded the operation with such success that we went to the printer a day ahead of schedule. He hired and patiently supervised the five secretaries who, in two very long weeks, sitting at NOVAR and IBM 2741 terminals, typed all the papers directly into computer files for subsequent editing. We are especially indebted to one of those secretaries, Susan S. Perkins, who is the only 150 baud typist known to me. Finally, Kent for another week babysat the terminal (a NOVAR fitted with a 'Delegate' type ball) while the Dartmouth RUNOFF system, running under time-sharing, generated properly formatted and paginated camera-ready copy. He joins me in thanking authors of papers for their promptness in returning corrections and supplying supplementary materials when necessary. Similarly we are grateful to panelists for submitting abstracts of position papers in advance for inclusion in the Proceedings. Happily, the printer, Dartmouth Printing Company, Hanover, N.H. (no relation), was able to complete the press run well ahead of his schedule. Kent and I, who are parvenues in the publishing game, wish to express a sense of scandalous indebtedness to the wisdom and years of experience of Victor G. F. Reynolds, Director of the newly established University Press of New England, Hanover, N.H. Graphical design of the Proceedings, as well as Conference programs, symbols and banners is due to John R. Scotford, Jr., Dartmouth College Designer. Art work included in the Proceedings was culled from contributions to the 1969, 1970, and 1971 Dartmouth Computer Art Contest.

No show is complete without an audience. Our man in the box office was Gregory S. Prince, Jr., whose Office of Summer Programs staff cut its big-conference teeth on the present one, apparently the largest meeting Dartmouth has ever hosted. Greg spent numerous sixty-hour weeks negotiating a transportation plan, handling registrations, making local arrangements, and publicizing the Conference. It appears likely that Greg distributed more literature at the Spring Joint Computer Conference than did the hardware merchants.

A final acknowledgment is due the audience itself--the growing number of undergraduate teachers who are making use of computers and computer-based methods in their courses.

Arth W. Lueh

Arthur W. Luehrmann
Proceedings Editor

TABLE OF CONTENTS

Page

UNDERGRADUATE INSTRUCTION (W9)

Chairman: Arthur Melmed — National Science Foundation

Dartmouth Project COEXIST--the Computer qua Computer — 1
Arthur Luehrmann — Dartmouth College

The Computer in a Responsive Learning Environment--Let a Thousand Flowers Bloom — 4
Alfred Bork — Univ. of California, Irvine

The Computer as a Means Toward Controversial Educational Ends — 10
Victor Bunderson — University of Texas

CALCULUS (W11A)

Chairman: Thomas M. Gallie, Jr. — Duke University

Calculus as a Laboratory Course — 13
L. Carl Leinbach — Gettysburg College

The Computer and the Calculus — 18
Theodore J. Benac — U.S. Naval Academy

Calculus and the Computer — 23
J.C. Mathews — Iowa State University

Computer (Programmable Calculator) Supplemented Ordinary Differential Equations — 27
Gene A. Kemper — Univ. of North Dakota

HUMANITIES (W11B)

Chairman: Sally Sedelow — University of Kansas

Building a Machine-Readable Text Library: Some of the Variables — 41
Ben R. Schneider, Jr. — Lawrence University

Computer-Aided Analysis of Music Used in Undergraduate Music Theory — 46
R.N. Killam — Tarleton State College

Training the Humanist for his Role in Developing Undergraduate Computer Oriented Courses — 50
Floyd R. Horowitz — University of Kansas

PHYSICS--COURSE IMPROVEMENT (W2A)

Chairman: John Merrill — Dartmouth College

A Computer-Based Introductory Physics Course — 51
Ronald Blum — Commission on College Physics, Univ. of Maryland

Using a Computer Instead of Calculus in an Introductory Physics Course — 58
Donald L. Shirer — Valparaiso University

Page

A Physics Course Based on a Set of Problems Solved by Computer — 64
Robert Ehrlich — State Univ. College

Computer Investigation of Normal Modes of Vibration for Coupled Oscillators — 68
Herbert D. Peckham — Gavilan College

ECONOMICS (W2B)

Chairman: Robert C. Bushnell — Ohio State University

Interactive Simulation Games in Economics — 75
John A. Naylor and William M. Baird — The College of Wooster

The DYNSIM System--A System for Student Investigation of Economic Models — 82
Paul N. Rappoport — Ohio State University

Evaluation of Student Learning and Information Utilization in a Computer-Simulated Economic Model — 94
Jo-Anne Sharac and Darlene Russ — University of Michigan

CHEMISTRY-I (W4A)

Chairman: Joyce H. Corrington — Xavier Univ. of Louisiana

Teaching Programming Languages and Optimizing Non-interactive Computer Applications (NICA) in the Chemistry Curriculum — 99
Ronald W. Collins — Eastern Michigan Univ.

Direct Classroom Use of a Time-Sharing Computer in Chemical Education — 112
Philip E. Stevenson — Worcester Poly. Inst.

The Use of APL in the Undergraduate Chemistry Curriculum — 117
Thomas R. Dehner and Bruce E. Norcross — State Univ. of New York, Binghamton

Computers and Chemical Education at Pitt — 130
K. Jeffrey Johnson — Univ. of Pittsburgh

SOCIAL SCIENCE--SOFTWARE SYSTEMS (W4B)

Chairman: Hugh Cline — Russell Sage Foundation

A Survey of Application Software for Social Data Analysis Instruction — 135
Ronald Anderson — University of Minnesota

Instant Turnaround and Conversational Computing as Instructional Tools in the Social Sciences — 142
Francis M. Sim, Laurence S. Rosen, Glen D. Kreider — Pennsylvania State Univ.

Computers and Undergraduate Training in an Underdeveloped Area: the Case of Louisiana State University in New Orleans — 152
John K. Wildgen — Louisiana State Univ.

We Don't Know What We Are Doing — 159
Edmund D. Meyers — Dartmouth College

Page

BIOLOGY (T9A)

Chairman: James D. Spain — Michigan Tech. Univ.

Uses of Computers in the Teaching of Ecology 171
Robert W. Gill, Stephen H. Bryant, Terry F. Mellon — Univ. of California, Riverside

The Computer in Undergraduate Genetics 176
H. Malcolm Owen — University of the South

Applications of a Wang Calculating/Computing System in the Undergraduate Genetics and Biometry Laboratory Instructions 181
J.C. Tan — Valparaiso University

Computers in the Undergraduate Curriculum in the School of the Allied Health Professions 187
Marion J. Ball — Temple University

COMPUTER PRODUCTION OF EDUCATIONAL MEDIA (T9B)

Chairman: Ronald Blum — Commission on College Physics

A General Graphics System for Computer Generation of Visual Aids 195
Tim G. Kelley — Oregon State University

Computer Animated Films With a Small Computer and a Small Budget 211
Bruce R. Thomas — Carleton College

Computer Animation for Chemical Education 214
Fred E. Heidrich and Kent R. Wilson — Univ. of California, San Diego

FINITE MATHEMATICS AND PROBABILITY (T11A)

Chairman: Barbara Beechler — Pitzer College

The Uses of Computing in a Modernized Probability and Statistics Course 217
Edward L. Spitznagel, Jr. — Washington University

Computing in Probability of Course 221
J. Laurie Snell — Dartmouth College

The Computer and Finite Mathematics 230
J. Richard VandeVelde, S.J. — Loyola University

Computer Supplemented Finite Algebra 238
James F. Peters — St. Johns University

DESIGN OF A COURSE IN SOCIAL APPLICATIONS OF COMPUTING (T11B)

Chairman: Andrew Molnar — National Science Foundation

Design of a Course in Social Applications of Computing 251
Andrew R. Molnar — Nat'l Science Foundation

Planning an Experimental Seminar on 'Computers, Society, and the Future' 253
Craig Decker — University of Virginia

Page

Course Presentation Units on Social Impact of Computers 263
Ronald E. Anderson — University of Minnesota

Social Implications of Computers at Scripps College 266
Lois Langland and F.W. Weingarten — Scripps College

Social Applications of Computing 267
Thomas E. Kurtz — Dartmouth College

THE MINI-COMPUTER IN THE LABORATORY (T2A)

Chairman: David Harris — Univ. of California, Santa Barbara

An Educational Mini-computer for the Laboratory 269
Charles L. Wilkens — University of Nebraska
Charles E. Klopfenstein — University of Oregon

The Introduction of Digital Computer Technology into the Undergraduate Chemistry Laboratory 277
S.P. Perone, F.E. Lytle, J.F. Eagleston — Purdue University

Computerized Experiments Using an A/D Converter 289
John H. Karl — Wisconsin State Univ.

Some Student Laboratory Experiments in Data Sampling by Mini-computer 299
Samuel A. Elder — U.S. Naval Academy

BUSINESS AND ACCOUNTING (T2B)

Chairman: George J. Brabb — University of Montana

Computer-Augmented Instruction in an Introductory Management Statistics Course 309
David P. Doane, Harvey A. Shapiro, Kenneth C. Young — Oakland University

Computer-Paced Instruction in the Business Curriculum 315
Thomas J. Dudley and Claude T. Ware — Pepperdine University

The Time-Sharing Computer as a Management Science Laboratory 322
J. Ronald Frazer — Clarkson College of Tech.

The Computer in Elementary Accounting 328
D.M. Ferguson — University of Minnesota

CHEMISTRY-II (T4A)

Chairman: Charles Klopfenstein — University of Oregon

Computer Simulation of Chromatographic Columns 339
Stephen Prager — University of Minnesota

ISIS--A Live Infra-Red Data System for Chemistry Education 347
Joseph R. Denk — N. Carolina Educational Computing Service

A Qualitative Organic Identification Game 359
Frederick M. Hornack — Univ. of North Carolina

Page

Computer Usage in Chemical Education at Massachusetts Institute of Technology 363
Paul L. Houston, Jeffrey I. Steinfeld, David G. Sutton — Mass. Inst. of Tech.

SOCIAL SCIENCES--COURSE IMPROVEMENT (T4B)

Chairman: Ronald E. Anderson — University of Minnesota

An Attempt to Utilize Computers in Urban Affairs Education 365
Brian C. Vargus and Douglas White — University of Pittsburgh

Computer Applications in the Undergraduate Sociology Curriculum at Oberlin College 371
Stephen J. Cutler and David V. Blagg — Oberlin College

Consortium Course: Computer Sciences, Statistics, and the Social and Managerial Sciences 379
David Milne, Paul Benson, George Weber — Bucknell University

Using the IMPRESS System to Teach Sociology 382
James A. Davis — Dartmouth College

ENGINEERING (F9A)

Chairman: James Monson — Harvey Mudd College

Introducing Optimizing Techniques into an Engineering Design Course 389
D.H. Offner — University of Illinois

Automated Integration of Subjective Criteria into Site Planning Models 398
David Morris — University of Virginia

Undergraduate Exercise Using a Computer Controlled Laboratory Model to Optimize a Temporary Cargo Storage Problem 411
Charles A. Moore — University of Illinois

PSYCHOLOGY (F9B)

Chairman: Lois Langland — Scripps College

Use of a Small Computer in an Undergraduate Psychological Laboratory: a Progress Report 417
David L. Sparks — University of Alabama

DATACALL: A Computer-Based Simulation Game for Teaching Strategy in Scientific Research 419
Richard Johnson — Earlham College

Computer Simulations in the Elementary Psychology Laboratory 424
Dana B. Main and Sabin Head — University of Michigan

Instruction in Statistics: A Report on the Computer Laboratory for Analysis of Data in Psychology 429
Arthur Sandusky, Francis Campos, James Livingston, Elijah P. Lovejoy, David M. Messick — Univ. of California, Santa Barbara

Page

ELEMENTARY STATISTICS (F11A)

Chairman: John W. Hamblen — Southern Regional Education Board

Use of Computers in Statistical Instruction — 437
Richard N. Schmidt — State Univ. of New York, Buffalo

Inclusion of Explanatory Material in Computer Programs for Statistical Analysis — 444
Jolayne Service — N. Carolina State Univ.

STATLAB, A Simple Programming System for the Statistics Laboratory — 456
G.F. Atkinson — University of Manitoba

PHYSICS--DIALOGS AND SIMULATIONS (F2A)

Chairman: Alfred Bork — Univ. of California, Irvine

Experience with a PLATO Mechanics Course — 463
Bruce Sherwood, Carol Bennett, Joan Mitchell, Charlene Tenczar — University of Illinois

Physics Computer Development Project--Computer Assistance in Student Problem Assignments — 469
Mark Monroe — Univ. of California, Irvine

Study of Waves Using an IBM 1130 Computer — 475
Paul A. Smith — Coe College

Introduction and Application of Computer Programming in Science and Technology Programs — 489
P. Hlavaty — Orange County Comm. Coll.

ACCOUNTING (F2B)

Chairman: George J. Brabb — University of Montana

The Time-Sharing Computer in the Business School Curriculum — 499
J. Peter Williamson — Dartmouth College

Computers and Computer-Augmented Accounting at Lenoir Rhyne College — 508
Craig C. Koontz — Lenoir Rhyne College

Computer-Augmented Accounting Education at Knox College — 516
Wilbur F. Pillsbury — Knox College

Other Submitted Papers — 523

Author Index — 529

DARTMOUTH PROJECT COEXIST--THE COMPUTER QUA COMPUTER

Arthur W. Luehrmann
Director, Project COMPUTe
Dartmouth College
Hanover, New Hampshire

All systems of classification are to an extent arbitrary. Labels and categories are not properties intrinsic to the thing itself, which stubbornly opposes pigeon-holing. Nevertheless, I have found it useful to divide the continuum of computer uses in education into the following ten modes:

1. Management of instruction and materials.
2. Administration of drill and practice sessions.
3. Conversations and dialogs.
4. Large data-base inquiry systems.
5. Simulation.
6. Problem solving.
7. Laboratory data analysis.
8. Laboratory data acquisition.
9. Control of experiments.
10. Production of graphics, movies and other media.

Among the papers presented at this year's conference and at those last year at the University of Iowa and at Illinois Institute of Technology one finds instances of most of these modes of use. Each has its enthusiasts, or at least its apologists and loyal defenders.

Items 1, 2, and 3 collectively constitute what most people appear to mean by Computer Assisted Instruction (CAI). In a complete CAI environment the computer would analyze the current state of the student's progress, would decide which materials or exercises to offer him next, and would administer most or all of the exercises, diagnosing responses to drills or dialogs, offering remedial help, giving pretests and posttests, and keeping detailed records. Most practical CAI work has centered on single components of such a hypothetical complete system. What we have today, with a few notable exceptions, are course fragments rather than stand-alone courses.

The report of the 1965 Irvine Conference (1) drew a single dichotomy (life was simpler in those days) between a so-called "linguistic" mode of computer use and a "computational" mode. Categories 1, 2, and 3 fairly well represent the linguistic mode. Computational mode referred to the processing of numerical information, rather than strings of characters, and it is represented in the present scheme by categories 4, 5, 6, and 7.

Large data-base inquiry systems, such as Meyers' IMPRESS (2) and Denk's ISIS (3), emphasize machine retrieval and processing of data. Numerical computation is the backbone of any simulation; the computer is programmed to exercise the mathematical model of a theoretical or actual system. Typically the student is not expected to understand the mathematical or programming details of the simulation, though of course he must believe that it is faithful to the system it represents. The student interacts with a simulation much in the way that he interacts with a real-world experiment, except that the knobs and dials are replaced by numerical parameters supplied to the program. The problem-solving mode differs most from simulations and data-base inquiry systems in that the student is expected to think algorithmically and to cast his problem in a form suitable for computer analysis. He must understand the underlying mathematics and the programming details. Laboratory data analysis is the remaining category with a principally computational character; though if the student merely exercises canned least-squares routines, the role of the computer may be unclear to him, and this category should be grouped together with the next two.

As computers become used more often as a common piece of laboratory apparatus it seems proper to identify categories 8 and 9 as distinct from previous categories. Automatic acquisition of experimental data and control of experimental processes are now a part of experimental science, and science education must include them.

Finally, it is becoming clear that graphical output from a computer calculation can convey far more and more valuable information than several yards of printer output. Category 10 includes real-time displays, slides, motion pictures, sound tracks, and any other media that may become appropriate.

The ten modes of educational computer use identified here are intended to be useful for purposes of discussion. Not every case will fit neatly into a single category. Most of the dialogs emanating from the Irvine Physics Computer Development Project (4,5), for example, have a large mathematical or computational component. It also frequently happens that a simulation will be imbedded in a highly conversational context with all the flavor of CAI. But even in these cases it seems possible to describe specific examples in terms of these categories.

BONES OF CONTENTION

It is the basic contention of this paper that, for the undergraduate curriculum in mathematics, physics, and engineering and most other sciences, the maximum return on a modest investment of developmental effort comes from work in categories 6, 5, and 7, in that order. Furthermore, I contend that work in these computational uses leads to a qualitative improvement in the teaching of the essential subject matter of these courses, in that it makes some central concepts possible to teach that were not possible or very difficult before. Third, I contend that these three categories of use develop in the student a sense of mastery of a powerful general tool. And finally, I contend that these uses, which are student-dominated rather than computer-dominated, are more in keeping with the liberal tradition of enquiry.

The first item in this contentious list is essentially economic: relatively small investment yields large educational dividends. Category 6 uses, for example, require only that the student learn how to solve problems via computer and that the teacher learn what problems are appropriate. Because this kind of use is "adjunctive" and not "mainline", to invoke Bunderson's dichotomy (6), it is both easier and safer to use. The designer is not responsible for programming the entire pedagogical strategy. Instead he looks for salient points in his course which might yield to an algorithmic approach or a simulation. By contrast, the CAI designer assumes full pedagogical responsibility. I do not contend, it should be noted, that CAI should not be developed, nor that in the long run it will not become cost-effective. But it does seem that the initial start-up costs are enormous. It probably requires a factor of ten or twenty times as much creative effort to cover in the CAI medium the same material which is covered in a textbook on the same subject. Even so, for many stable points in the curriculum the investment is amply recoverable.

While on the economic issue, it is worth calling attention to the fact that, given the reward structure of academia, a teacher who develops CAI materials and turns over a portion of the pedagogical responsibility to the computer, is usually acting against his self-interest. I know of no one whose course load was reduced because of his CAI results. In fact the majority of my acquaintances in this field have lost their jobs. If CAI is to be developed on a grand scale it must, in my opinion, be done by organizations which operate independently of departmental and professional stresses. But for this reason such organizations will have difficulty attracting good, seasoned teachers to do the creative work. That is why most CAI work emanating from the computer industry is so uniformly bad. What is probably required is separate CAI organizations within universities--organizations that can draw on the faculty there, and can reward them by other than professional means.

My second contention regarding computational mode uses is that they lead to a qualitative improvement in the teaching of the essential subject matter of mathematics and science courses. One does not merely do more of the same, but one is able to teach altogether new things. The initial-value problem for systems of coupled second-order non-linear differential equations, for example, can be understood by college freshmen in about a week, thanks to an algorithmic approach and the availability of a computer to exercise the algorithms. Normally this subject is never approached formally at all, though it is a crucial problem in all the mathematical sciences. It is this sort of curricular application which has highest priority in Dartmouth Project COEXIST (Computer Oriented Experiment in Science Teaching, work supported by a grant from the National Science Foundation). A number of examples of this kind will be presented in the talk.

The third claim regarding the computational mode is that it gives the student mastery of a powerful general tool. When he succeeds in solving an otherwise intractable problem by means of a computer program, he becomes aware of the power and usefulness of the method. Writing a computational program constitutes an enhancement of professional competence that is of a far higher order than merely being the subject of a CAI exercise. It may be that the struggle for improvement and the sense of individual mastery is the source of student involvement with computing.

The final claim is related to this. Computational uses are learner-dominated; CAI is teacher dominated (though vastly less so than most lectures, for example, are). When a student is solving problems by computer, or processing laboratory data, or interrogating a large data base he is in charge of the situation. He decides what is to be done and he directs the action. The computer and its software represent an intellectual resource, like a library or an army of research assistants. The student commands the resource according to his perceived needs. The CAI ambience is quite different. Taylor (7) reports on CAI work done by a group of physicists using the very flexible ELIZA system to write dialogs on special relativity. He observed that the students appeared to feel themselves subjects in a somewhat submissive role, rather than masters of a new medium. If this is true generally, it raises questions regarding the place of CAI in the tradition of liberal learning. Can a CAI dialog attain the Socratic ideal? But then it is fair to ask, how many warm-bodied teachers succeed in that objective? It may well be that the CAI medium is improvable beyond the average teacher. But let us not confuse the issue. What is claimed here is that the student who is engaged in applying the computer resource to solve a problem is in a less servile, hence, more desirable, posture than a student who is in the clutches of either a CAI program or a typical lecturer. The chief advantage of the computer lies in the fact that large amounts of the curriculum can be restructured away from the lecture and toward a problem-solving context. This restructuring has always been possible of course; but now the computer permits the student to see actual solutions to a large number of interesting and realistic problems.

The talk will present examples of work in most of the ten categories defined here, in order to give specific meaning to some of these ideas.

REFERENCES

(1) "The Computer in Physics Instruction--Report on the Conference on Uses of the Computer in Undergraduate Physics Instruction, November 4-6, 1965, at the University of California, Irvine. Published by the Commission on College Physics, University of Maryland, College Park, Md.

(2) Davis, James A. and Sternick, Joanna H., "The IMPRESS Primer," Dartmouth College, Hanover, N.H., February, 1971 (2nd printing) multilithed.

(3) Denk, Joseph R., "ISIS--A Live Infra-Red Data System for Chemistry Education," Proceedings of Conference on Computers in the Undergraduate Curricula, Dartmouth College, Hanover, N.H., June, 1971.

(4) Bork, Alfred M., "The Computer in a Responsive Learning Environment--Let a Thousand Flowers Bloom," Proceedings of Conference on Computers in the Undergraduate Curricula, Dartmouth College, Hanover, N.H., June, 1971.

(5) Monroe, Mark, "Physics Computer Development Project-Computer Assistance in Student Problem Assignments," Proceedings of Conference on Computers in the Undergraduate Curricula, Dartmouth College, Hanover, N.H., June 1971.

(6) Bunderson, Victor, "The Computer as a Means Toward Controversial Educational Ends," Proceedings of Conference on Computers in the Undergraduate Curricula, Dartmouth College, Hanover, N.H., June, 1971.

(7) Taylor, Edwin F., "Automated Tutoring and its Discontents," Amer. J. Phys. 36, 496 (1968).

THE COMPUTER IN A RESPONSIVE LEARNING ENVIRONMENT--LET A THOUSAND FLOWERS BLOOM

Alfred M. Bork
University of California
Irvine, California

My tactics in my brief time will be as follows: First, like Martin Luther, I will tell you my basic theses. Second, I will illustrate them by describing a course taught the past three years, and finally I will briefly restate my position.

Thesis I: Different students learn in different ways.

Most teachers would agree, but practically all courses seem to ignore student differences. Most courses are rigidly structured, with only one path to success and only one set of learning materials. An environment truly responsive to students must have a variety of materials and techniques for learning.

Thesis II: We are only beginning the task of learning how to use computers in education.

I worry greatly about teachers who feel that they already know all the answers. We have a long way to go, and theoretical analysis will not tell us how to employ computers effectively. Hence we want to maintain flexibility, and we should be prepared for long years of trial and error while using computers in learning.

Thesis III: Useful ways to involve computers in teaching may depend on subject matter involved.

What is highly effective in physics may turn out to be useless for literature. This is obvious with computational uses, which are tailored for a specific need. It is perhaps less obvious with other types of usage. While we may find techniques which transcend subject matter boundaries, we will continue to need specific techniques for individual areas.

Thesis IV: It is wise to retain all usage modes for computers in every learning situation.

Computational, tutorial, simulatorial, managerial, and other modes as yet unnamed may all prove to be of great importance in education. Further, in view of Thesis III, a mode worthless in one discipline may be valuable in another area.

Thesis V: We should continue to develop ways of learning independent of computers.

Some areas and some students may profit from other techniques. Many powerful learning tools exist. The film can be used interactively with computers as a very effective learning medium. Furthermore, in some areas, nothing competes in teaching effectiveness with a student's experience in working problems. Thus, I am not in sympathy with the "whole course" approach to computers in education, where the computer becomes the sole medium. Today we don't know enough to use computers exclusively, and we may never want to.

Thesis VI: The test of all learning is with students--does the material, computer or otherwise, lead to some type of learning for some students?

All educational materials need to be widely used and continually modified on the basis of students' experience.

So much for my theses, which I now hope to illustrate. The course I am about to describe is only a first approach to these ideas. It comprises the first two quarters of the five quarter beginning science and engineering majors' physics course at Irvine. It is, like much else in educational computer usage, an early endeavor, one that I suspect will seem crude when we finally learn how to use computers effectively. The average enrollment has been 160 students.

At the beginning the student is given 20 weekly assignment sheets. Each sheet specifies the subject area covered and the learning devices and modes available for that topic. A typical weekly sheet is shown at the end of the paper. The course has a text (prepared at Irvine), and each week's assignment suggests something to study from the text and from references. Often The Feynman Lectures are referenced to go along with the week's assignment, and also references are made to books at other levels, to accommodate the variety of students.

This course, like many physics courses, is oriented toward the problems. Assignments contain both required problems and optional problems. The computational aspects of using the computer make themselves felt here; some of the problems can be handled only by means of the computer. The overall effect of the computation mode on the course is tremendous, affecting the type and level of material the student is capable of dealing with. Thus even the beginning student can immediately start using the laws of motion as differential equations and can study the behavior of many physical systems. I will not describe this aspect further as I have already done so a number of times.

Films are used in three ways. First, we have two loop boxes, which show a loop on a rear screen projection system each time a button is pressed. Films in these boxes appropriate to the course are announced on the weekly assignment sheet. One "lecture" period per week is used for long films. These films do not necessarily tie closely with the material of the week, but are often motivational. Finally, short films are occasionally used in lecture when they fit in dynamically with what is happening.

The student is encouraged to think of the two lectures each week as only one of the learning devices in the course; his attendance presumably depends on how effective he finds the lecture as an aid to working the problems. (Most students come to most lectures.) We do not believe, from querying our students, that students are willing to give up lectures entirely at present. Whether or not they are effective learning devices, students still want them.

The aspect I will discuss in most detail is the student-computer conversation, the dialogs, available continuously during the quarter. We have wondered about where within physics, and with what kinds of dialogs, we could hope to get extra leverage in teaching. Therefore we have not tried to computerize everything, trying always to be highly selective.

One type of dialog we have stressed can be described as the interactive proof dialog. These dialogs can be used by students as replacements for, or supplements to, lectures. The main objective is to involve the student in proving the "main line", often difficult, results on which the course depends, changing these proofs from the passive experience they are for many students in lecture and reading into an active experience where the student tries to make the critical steps, first alone, then with increasing assistance. An example of a student use of a dialog from the second quarter appears at the end of this paper.

Another type of dialog is being described by Mark Monroe at this meeting. We try to assist the student with a problem he has attempted unsuccessfully. As indicated, the course is problem-oriented; week after week the students face the task of working problems difficult for them. To develop the art of working such problems is the main goal, rather than to yield any particular physical information. The student who has direct difficulty with the problem can seek out the instructor, but in a large course this may not be feasible at the moment when he needs assistance. The dialogs can, hopefully, aid him. The easiest time to write such problem assistance dialogs is just after the problem has been assigned in class.

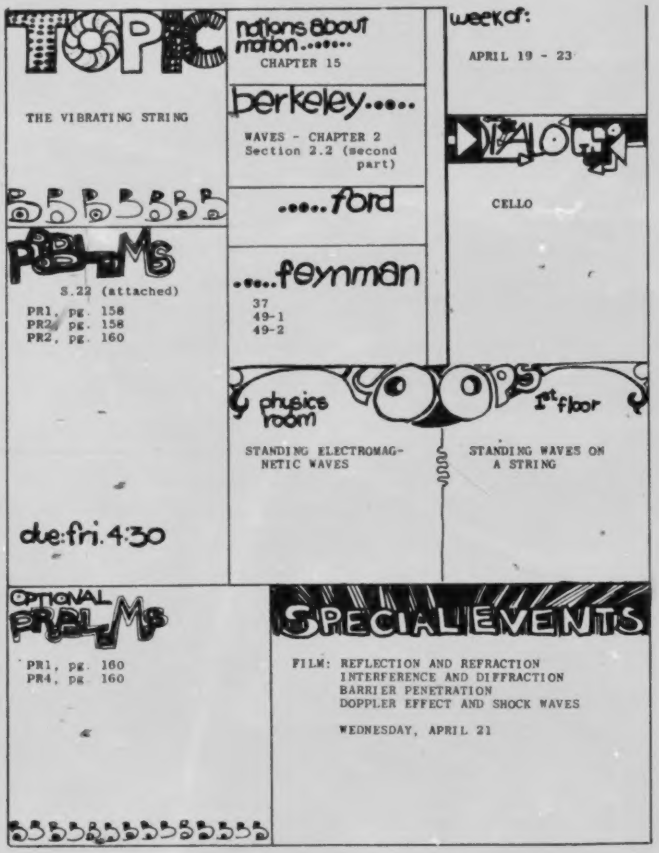
A third use of dialogs is for remedial mathematical assistance. The mathematical limitations of students often present a major pedagogical problem. Thus when we want to use complex numbers, we are often hindered by the very mixed background of the class. Furthermore those who have had exposure may need review. One doesn't know whether to lecture, and bore half the class, or assign reading which might not match the level of some. The diagnostic-remedial dialog such as our complex number dialog tries to determine what the student doesn't know, and then it assists him.

The fourth type of dialog used in the course is a simulation of a physical system; this year we have had two examples. One was a simulation of a lunar landing, from friends at Berkeley (Steve Derenzo and Noah Sherman). Knowledge about motion at constant acceleration is sufficient to provide a successful landing. Variants of this program are common. A second simulation is, from a pedagogical point of view, more ambitious because it attempts to teach, through a measurement-type discovery procedure, an important property of waves, the travelling pattern idea. It was developed jointly by myself and John Robson of the University of Arizona. A small part with student output underlined appears at the end of this paper.

Dialogs when first written, even by the best of instructors, often turn out to be poor; none of us is good enough to anticipate all the right responses, all the reasonable wrong responses we should respond to, etc. So as the student is using the dialog, we file responses we cannot analyze and other information about his progress which may help us to improve the dialog for future students. Thus gradually we make the dialogs more effective through extensive use with students and through successive revisions.

Since the dialogs are not required--like the lectures, they are used by students who they feel they help with problems--we also want to find which students are using which dialogs, and whether this use affects their performance. In spite of the large effort required one must always remember that the dialog might turn out to be useless, so that no amount of improvement through usage will produce a viable teaching program. Edwin Taylor at MIT has suggested that one way of telling what is useful is by letting students decide. Hence our system records dialog usage.

This is just one course. I hope that many other courses in physics, and in many other areas which use computers, will be developed. I believe that we should have a thousand flowers blooming. I realize that many of them, probably even some of my own, will be ugly. The botanist knows that not all flowers end up in elegant, cultivated gardens. Only through careful breeding over many years can we find the desirable types. And in our responsive learning environment, an environment responsive to student needs, many flowers besides the computer flowers will also be blooming.



TOPIC

THE VIBRATING STRING

notions about motion.......

CHAPTER 15

berkeley.....

WAVES - CHAPTER 2
Section 2.2 (second part)

.....ford

.....feynman

37
49-1
49-2

week of:

APRIL 19 - 23

DIALOGS

CELLO

PROBLEMS

S.22 (attached)

PR1, pg. 158
PR2, pg. 158
PR2, pg. 160

due: fri. 4:30

physics room

OOPS

1st floor

STANDING ELECTROMAGNETIC WAVES

STANDING WAVES ON A STRING

OPTIONAL PROBLEMS

PR1, pg. 160
PR4, pg. 160

SPECIAL EVENTS

FILM: REFLECTION AND REFRACTION
INTERFERENCE AND DIFFRACTION
BARRIER PENETRATION
DOPPLER EFFECT AND SHOCK WAVES

WEDNESDAY, APRIL 21

```
GET DIALOG
    - PROGRAM NAME? CELLO

PLEASE TYPE AN IDENTIFICATION
- 7 CHARACTERS OR LESS - - - ALFRED

THAT IDENTIFICATION HAS BEEN USED BEFORE.
WAS IT BY YOU?

?NO

THEN PLEASE INPUT ANOTHER IDENTIFICATION.
- 7 CHARACTERS OR LESS - - - AND

FIRST, LET'S TALK SOME ABOUT THE
WAVE EQUATION IN ONE DIMENSION.
WHAT KIND OF EQUATION IS THE WAVE
EQUATION?

?A LINEAR DIFFERENTIAL EQUATION

WHAT KIND OF DIFFERENTIAL EQUATION?

?PARTIAL

YES, IT'S A PARTIAL DIFFERENTIAL
EQUATION.  GOOD.
WHAT IS THE ORDER OF THE DERIVATIVE
WITH RESPECT TO POSITION?  IS IT A
FIRST DERIVATIVE, OR WHAT?

?SECOND

RIGHT.  IT IS A SECOND DERIVATIVE.
WHAT ABOUT THE ORDER OF THE TIME DERIVATIVE?

?2ND ALSO

YES, IT IS ALSO A SECOND DERIVATIVE.
IN ADDITION TO DERIVATIVES, THE WAVE
EQUATION CONTAINS ANOTHER PHYSICAL
QUANTITY ASSOCIATED WITH WAVES.
WHAT IS IT?

?FREQUENCY?

THIS IS A WAVE PROPERTY, BUT ONLY
FOR CERTAIN TYPES OF WAVES.  WHAT
OTHER QUANTITY OCCURS IN THE WAVE
EQUATION?

?THE VELOCITY OF THE WAVE

YES, THE WAVE EQUATION CONTAINS
THE WAVE VELOCITY.
```

⋮

```
WE WANT SOLUTIONS OF THE WAVE EQUATION
FOR A STRING WITH FIXED ENDS, FIRST FOR
MOTIONS IN WHICH ALL PARTS OF THE STRING
MOVE WITH THE SAME FREQUENCY. SO
WHAT SHOULD WE ASSUME FOR THE FUNCTIONAL
DEPENDENCE OF THE SOLUTION ON TIME?

?SIN OR COS

THERE ARE THREE POSSIBILITIES HERE FOR
THE TIME DEPENDENT PART OF THE SOLUTION,
     SIN WT, COS WT, AND EXP(IWT)
```

```
THE 'NORMAL MODE' SOLUTIONS OF THE
WAVE EQUATION ARE OF THE FORM
     A(X) EXP(IWT) OR A(X) COS WT OR A(X) SIN WT
THE POSITION X SHOWS WHERE WE ARE ALONG THE STRING.
YOU SHOULD CONTRAST THIS
WITH THE TWO-BODY NORMAL MODE SOLUTION,
THE FUNCTIONS REPRESENTING THE MOTION OF
THE TWO MASSES ARE:
     X1=A EXP(IWT) AND  X2=B EXP(IWT)

WHAT IS THE PHYSICAL MEANING OF A(X)

?THE AMPLITUDE

YES.....
ALL POINTS ON THE STRING OSCILLATE WITH THE
SAME FREQUENCY, BUT WITH AN AMPLITUDE WHICH
VARIES WITH POSITION ALONG THE STRING.

WHAT DIFFERENTIAL EQUATION MUST BE SATISFIED BY
THE AMPLITUDE   A(X)

?THE WAVE EQUATION

THE FULL EXPRESSION
     A (X)   EXP(IWT)
SATISFIES THE WAVE EQUATION. WRITE THE
EQUATION A SATISFIES.

?  D↑2 A / DT↑2 = - (W/V)↑2 * A

TROUBLE RECOGNIZING YOUR RESPONSE
PERHAPS WITH THE CLUMSY DERIVATIVE
NOTATION NEEDED AT A TYPEWRITER.
INDICATE THE FIRST DERIVATIVE WITH RESPECT TO
X BY A' AND THE SECOND DERIVATIVE BY
     A''

?   A'' = -(W/V)↑2 A


SO A'' + (W/V)↑2 A = 0

THIS EQUATION MAY WELL LOOK FAMILIAR
TO YOU. WITHOUT GOING INTO PRECISE
DETAILS, DESCRIBE THE SOLUTIONS A(X).

?SINS AND COS

YES.
COMPLETE THIS SOLUTION:
     A(X)=SIN ( W/V T)

POSITION -X- IS THE INDEPENDENT VARIABLE.
COMPLETE THIS SOLUTION:
     A(X)=SIN ( W/V X )

FINE......
COSINES WOULD ALSO WORK.
SO THE NORMAL MODES OF OUR VIBRATING
STRING ARE OF THE FORM
     SIN(W/V X) EXP(IWT)
OR, WORKING WITH THE REAL PART,
          SIN (W/V X) .COS WT

BUT WE ARE STILL NEGLECTING PART OF THE
ORIGINAL STATEMENT OF THE PROBLEM. DO
YOU REMEMBER WHAT ASPECT OF THE
VIBRATING STRING, AS ORIGINALLY DESCRIBED
HAS NOT BEEN USED HERE YET?

?NO

IT WAS A LONG TIME AGO!

WE INITIALLY REQUIRED THAT THE ENDS BE
FIXED, NOT MOVING, AND WE HAVE NOT MADE
USE OF THIS INFORMATION.

CAN YOU SEE WHICH QUANTITY IN
OUR NORMAL MODE SOLUTION,
          SIN (W/V X) .COS WT
WILL BE RESTRICTED BY THE CONDITION
OF FIXED ENDS?
```

```
!GET DIALOG
    - PROGRAM NAME? ROPEGAME

THE PHYSICAL SYSTEM WE WILL EXPLORE IS
AN EXTREMELY LONG ROPE WITH A DISTURBANCE
IN IT IF YOU TELL ME A POSITION ALONG THE ROPE
AND A TIME, I WILL GIVE YOU THE DISTURBANCE,
THE DISPLACEMENT FROM EQUILIBRIUM. YOUR JOB
IS TO LEARN WHAT IS HAPPENING IN THE ROPE.

I WILL EVENTUALLY TURN THE TABLES, GIVING YOU
INFORMATION AND ASKING YOU TO PREDICT VALUES.

POSITION IS IN METERS AND TIME IN SECONDS;
DON'T ENTER UNITS.


TIME = 2            POSITION = 2            DISTURBANCE =   0
TIME = 5            POSITION = 5            DISTURBANCE =   0
TIME = 58           POSITION = 475          DISTURBANCE =   0
TIME = 0            POSITION = 0            DISTURBANCE =   0.28
TIME = -1           POSITION = -1           DISTURBANCE =   0

YOU'VE FOUND THE ACTION. KEEP GOING....

GRAPHS OR SKETCHES MIGHT BE USEFUL.
TIME = 1            POSITION = 1            DISTURBANCE =   0
TIME = 1            POSITION = 3            DISTURBANCE =   0
TIME = 1            POSITION = 5            DISTURBANCE =   0.15
TIME = 1            POSITION = 6            DISTURBANCE =   0
TIME = 1            POSITION = 3.5          DISTURBANCE =   0.08

THIS PUZZLE HAS A 'PAYOFF'. IF YOU
CAN DETERMINE HOW THIS DISTURBANCE BEHAVES,
YOU WILL UNDERSTAND AN IMPORTANT PRINCIPLE
INVOLVED IN MANY PHYSICAL SYSTEMS.
AFTER A FEW MORE MEASUREMENTS YOU CAN
TURN-THE-TABLE AND TRY TO PREDICT THE
BEHAVIOR OF THE ROPE.
TIME = 1            POSITION = 4.5          DISTURBANCE =   0.15
TIME = 0            POSITION = 5.5          DISTURBANCE =   0
TIME = 1            POSITION = 4            DISTURBANCE =   0.28
TIME = 1            POSITION = 4.75         DISTURBANCE =   0.15
TIME = 1            POSITION = 5.25         DISTURBANCE =   0.08

YOU MAY HAVE SOME IDEA OF HOW THE ROPE IS
BEHAVING. AT THIS POINT I WILL CHANGE THE
RULES OF THE GAME. FOR:
     MEASUREMENT  TYPE M
     TURN-THE-TABLES TYPE T
     LIST OF MEASUREMENTS  TYPE L
     GRAPH    TYPE G
DON'T BE DISTURBED IF YOU CAN'T TURN THE
TABLES AT FIRST--I WILL GIVE YOU OTHER CHANCES,
MEASURE? LIST? TURN? GRAPH?  G

WHAT INDEPENDENT VARIABLE DO YOU WANT FOR
YOUR GRAPH?   POSITION
```

(Program continued on the next page)

```
FOR WHAT VALUE OF TIME? 1
     MIN     HORIZONTAL      MAX      6
 0     MIN      VERTICAL       MAX       0.2755
---------------------------------------+--------------------------









                                                 +
                                             +      +




                                                        +
                                 +





+-------------------------+---------------------------------------+
MEASURE? LIST? TURN? GRAPH?   G

WHAT INDEPENDENT VARIABLE DO YOU WANT FOR
YOUR GRAPH?   X

FOR WHAT VALUE OF TIME? 0

NOT ENOUGH MEASUREMENTS AT THAT VALUE TO
BOTHER PLOTTING.
MEASURE? LIST? TURN? GRAPH?   M

HOW MANY MEASUREMENTS IN THIS BLOCK?  5

TIME = 2             POSITION = 7              DISTURBANCE =    0
TIME = 2             POSITION = 8              DISTURBANCE =    0.28
TIME = 2             POSITION = 9              DISTURBANCE =    0.15
TIME = 0             POSITION = 7.5            DISTURBANCE =    0
TIME = 2             POSITION = 7.5            DISTURBANCE =    0.08
MEASURE? LIST? TURN? GRAPH?   T

YOU KNOW ALREADY THAT AT T =  0 AND AT  X =  0
     THE DISTURBANCE =     0.28


AT T =    2.99  THE DISPLACEMENT IS TO BE THE SAME.
WHAT VALUE OF POSITION MAKES THIS THE CASE?

?11.9

SEEMS GOOD. LET'S TRY ANOTHER OF THE
SAME TYPE.
YOU KNOW ALREADY THAT AT T =  1 AND AT  X =  5
     THE DISTURBANCE =     0.15


AT T =    5.14  THE DISPLACEMENT IS TO BE THE SAME.
WHAT VALUE OF POSITION MAKES THIS THE CASE?

?21.6

FINE.....NOW WE'LL PLAY THE GAME A SLIGHTLY
DIFFERENT WAY.
```

THE COMPUTER AS A MEANS TOWARD CONTROVERSIAL EDUCATIONAL ENDS

Victor Bunderson
University of Texas
Austin, Texas

ABSTRACT

At last year's Conference on Computers and the Undergraduate Curriculum, I made a distinction between "adjunct" and "mainline" approaches to the uses of computers in instruction.

In the first approach the computer is an adjunct to the teacher and to his regular classroom or laboratory. Authoring of adjunct materials requires only relatively minor changes in the teacher's role. Economically, adjunct instruction represents an add-on cost to present instructional systems, but brings increased effectiveness.

Mainline instruction, on the other hand, calls for revision of the entire system. The organization of instruction is changed, the grading procedures are changed, and the role of the teacher is changed. This new system envisions a modular organization for instructional units and a self-paced, individualized environment. The materials are engineered for effectiveness, efficiency, and appeal to the students. Such systems are capable of decentralization and broad dissemination since they are not dependent upon the presence and personality of particular individuals. These systems are designed to take over large blocks of instruction in high volume areas such as freshman mathematics and English, undergraduate computer science, and foreign language instruction. When mainline instruction is used, the roles of some teachers will change from that of presenters of information to managers of individualized systems. Other teachers will become well-paid designers of instructional programs and materials. Others, freed of administrative and presentation concerns, will function in the "Master Teacher" role, emphasizing human to human interaction on individual projects, and to motivate, guide, and inspire. Still others may be displaced.

The new profession of "courseware-designer" or "courseware-engineer" will include new skills such as those of behavioral analysis and the application of empirically derived principles of instruction. Such principles are forerunners of a theory of instruction and many already are non-trivial, non-intuitive, and empirically validated.

Economically, mainline instruction need not represent an add-on cost but is an alternative to labor-intensive, traditional modes. It can bring increased effectiveness and a substantial increase in efficiency.

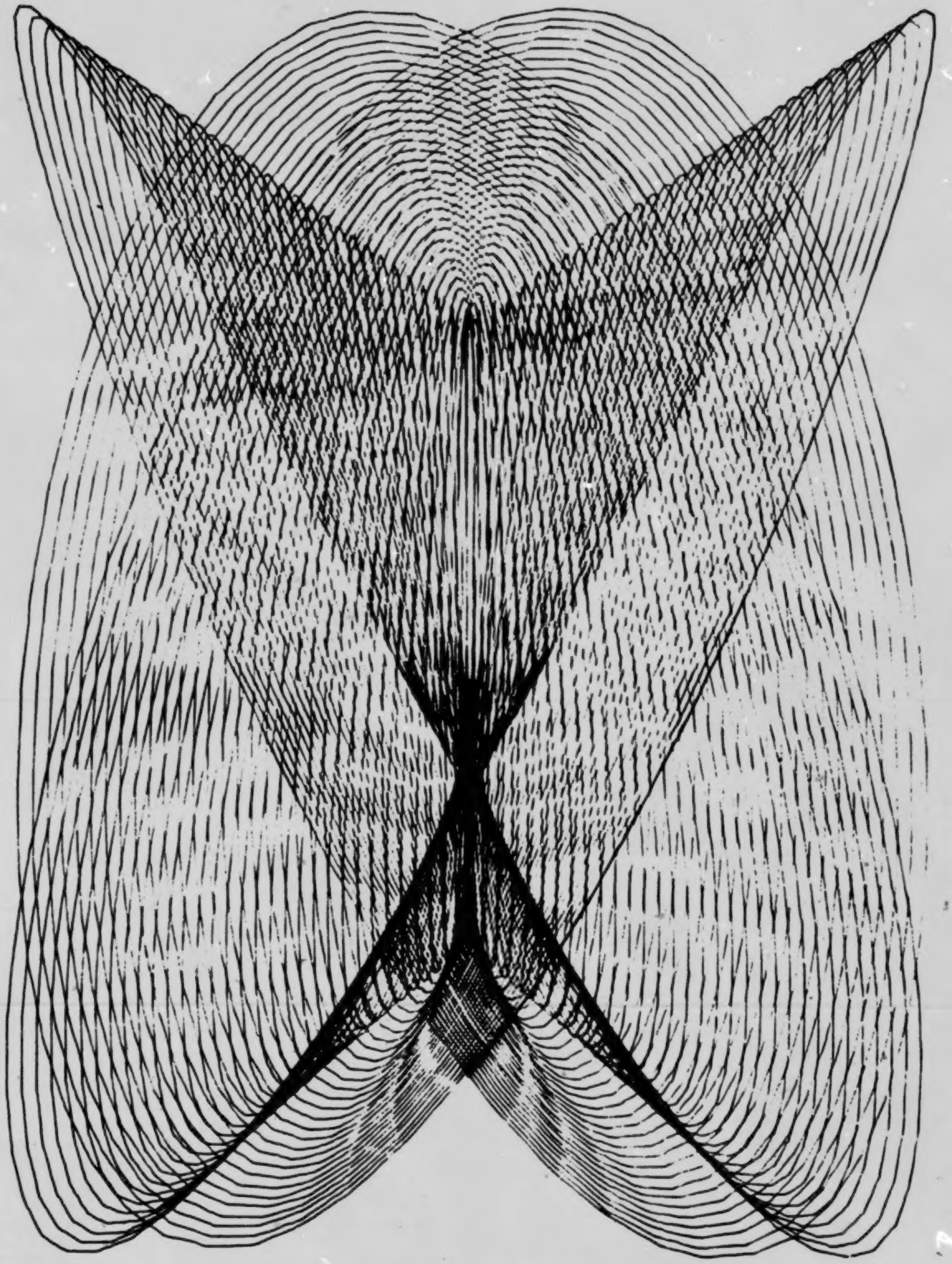
While the term "mainline instruction" has been useful to draw attention to the above distinction, it can be construed too narrowly. It is the purpose of this paper to broaden the concept now designated by that term and in so doing to reveal a strategy for educational reform in which the computer is a necessary tool, but not the central theme. In addition to discussing some of the needs for and some of the elements of this strategy, the function of the adjunct use of the computer in today's and tomorrow's educational systems is discussed.

Using data from a recent NSF-financed study of higher education, emerging problems which will dictate radical changes in higher education were discussed. The proposition was defended that extensions of present labor-intensive patterns for instruction cannot meet the coming crises in higher education. Incidental to this problem, it can be seen that the adjunct use of the computer, being an expensive additional cost, cannot be considered to be a solution to these critical problems. In addition, adjunct instruction depends heavily on individual instructors. It also emphasizes strategies which require rather bright and highly selected students. It therefore was not seen to constitute a direct solution to the problems of increased enrollment and reduced entrance requirements.

Mainline instruction, on the other hand, was seen to offer an economically attractive and educationally effective alternative.

The change from labor-intensive to technology-intensive educational systems naturally conveys Orwellian fears to the hearts of many teachers. A more optimistic picture was briefly sketched through a description of what mainline instructional systems could mean to students and to the profession of education. More control and responsibility can and should be placed in the hands of the students. Built-in mechanisms for growth and adaptation of materials must be a part of widely distributed mainline systems.

Contrary to the notion that mainline and adjunct uses of the computer are competitive, the widespread availability of mainline systems will permit greatly expanded use of the economical and powerful computer systems by teachers and students at all levels of education. It is important that this adjunct use be encouraged and expanded, for in addition to providing students and faculty with powerful new conceptual tools, it often leads to the creative restructuring of knowledge, and to the discovery of effective new instructional techniques. This innovation, in my view, is the most important benefit of adjunct CAI, and hopefully will provide a continual spring of fresh input into widely disseminated, cost-effective mainline systems.



Stephen Toll

CALCULUS AS A LABORATORY COURSE

L. Carl Leinbach
Gettysburg College
Gettysburg, Pennsylvania

Introduction: In considering the role of the computer in an undergraduate mathematics curriculum, one is faced with the task of considering the relationship between the disciplines of computer science and mathematics. Is the computer merely a tool to be used for illustrating mathematical algorithms and carrying out computations which are too tedious and time consuming by hand calculations? Certainly such possibilities are now available to us with the availability of high speed computers. Instead of computing one or two values for a Taylor series we can compute a whole table of values much more quickly and efficiently. Epsilons and deltas in various limit processes can be easily computed and processes can be illustrated much more clearly than was previously possible. On the other hand, should we change the type of mathematics we teach in order to make the use of the computer an integral part of the mathematics program? Many new calculus texts are stressing discrete methods and numerical techniques. Many of the theorems and techniques of mathematics serve as a useful source for computing algorithms. For many, the purpose of taking a calculus course is to develop techniques for problem solving.

G. J. Porter (3,4) has proposed the use of the computer as a device for the mathematics student to discover meaningful definitions and proofs of theorems. This use of the computer in the mathematics curriculum is refreshingly unique and, from the viewpoint of the pure mathematician, one of the most palatable uses of the computer; however, it seems to this author that such a use is best geared to upper division mathematics courses or small seminars in calculus, not the larger classes that we find ourselves confronted with in the normal teaching situation.

The viewpoint which the program at Gettysburg College adopted was essentially a middle ground between the two extremes expressed in the first paragraph. Such a course was implied by J. T. Schwartz in his essay on the "Prospects of Computer Science" in the Cosrims essays (5). In this essay Schwartz states:

> until the development of computers the possibility of dealing with the complex was never really envisaged. Perhaps the most successful substitute for such a possibility, as well as the nearest approach to it came in mathematics. Mathematics provided simplification in a different way: by furnishing intellectual techniques of conceptualizing abstraction and developing symbolic and nonsymbolic languages to facilitate the manipulation and application of concepts...To find the simple in the complex, the finite in the infinite--that is not a bad description of the aim and essence of mathematics....
>
> In this quest for simplification, mathematics stands to computer science as diamond mining to coal mining. The former is a search for gems. Although it may involve the preliminary handling of masses of raw material, it culminates in an exquisite item free of dross. The latter is permanently involved with bulldozing large masses of ore--extremely useful bulk material....

Mathematics and computer science are distinct and very different disciplines. Neither is in a position of being taken over or dominated by the other; however, neither dares continue to exist in total ignorance of the other. The advent of the computer has changed and will continue to change the nature of the problems which mathematicians are considering. Likewise advances in mathematics have and will undoubtedly continue to produce advances in computer science. In short, the relationship between mathematics and computer science is not unlike the relationship between mathematics and the natural sciences. The latter possesses interesting and challenging problems for the former, while advances in the former will stimulate and aid the advancement of the latter. It is for this reason that the program at Gettysburg College was felt to be essential and was instituted.

Background: In 1968 an IBM 1130 computer was installed on the campus as part of a COSIP grant from the National Science Foundation. The computer was an 8K memory, three disk drives and had available a Calcomp plotter and a 1132 printer.

At the time the computer arrived on campus the great majority of the faculty (including the author) and students had no experience or very little experience in computer programming.

During the 1968-1969 academic year most of the work done on the computer was of an elementary nature as the faculty and students became acquainted with programming techniques and some of the capabilities of the computer. The applications of the computer in academic work were at best superficial. The original grant included funds for educating both faculty and students in the use of the computer. This program was carried out during the Spring and Summer of 1969. The 1969-1970 academic year saw increased use of the computer in many disciplines. This past year the trend has continued. Not only is the computer becoming an integral part of the mathematics program, but also it is widely used by the physics, chemistry, economics, political science, psychology and sociology departments.

The Course: The freshman calculus course at Gettysburg consists of four sections of between 25 and 30 students. Each section meets four times during the week for 50 minute sessions. This arrangement usually guarantees that all students will have the fifth day free at that time slot allocated for their calculus course. This fifth day can be used as a meeting time for discussion of laboratory projects. Discussion of the computer projects was not, however, confined to this period. Whenever possible, the algorithms and techniques of the computer assignments were used to emphasize points in class, and in many cases the differences in techniques proved to be useful for illustrating the meaning and power of certain theorems. The text for the course is Calculus by Lipman Bers.

It was assumed that the students had no previous experience in computer programming. Consequently, the first four class meetings were used to introduce the student to algorithmic processes, flow charting and the basics of FORTRAN programming. These lectures began in the standard fashion with a discussion of the meaning of the term algorithm and some algorithmic processes in daily life (changing a flat tire, following a recipe, etc.). A flowchart for the process of changing a flat tire was developed and refined. The discussion was then directed to the Fibonacci sequence and a flowchart was developed for generating the terms of this sequence.

The second lecture provided an introduction to the very basics of FORTRAN: variable names, assignment of values, and reading values stored in the computer memory. Brief mention was made of the difference between integer and floating point modes of storage. A FORTRAN program for the Fibonacci sequence was written using the flowchart from the previous day. After writing the program the students and the instructor actually traced several steps of the program. This served to review the ideas behind the operations of assigning values to a variable and reading these values from memory. The discussion was concluded with a detailed card-by-card discussion of the method of keypunching the program. One of the assigned problems from this class period was to actually punch and run the Fibonacci sequence program.

Lectures three and four were devoted to integer and floating-point modes of storage and arithmetic in these modes, iterative processes (DO loops), arithmetic expressions, precedence levels for carrying out operations within an arithmetic expression and FORTRAN library functions.

In addition to introducing the basic concepts of FORTRAN programming, these first four lectures served to prepare the student for the calculus. The ideas of sequences and functions were introduced and discussed. The students were, in general, much more enthusiastic about these notions than is usually the case with the standard presentation of this material. During the discussion of DO loops, the standard summation notation was introduced. A comparison was made between the information required in a DO statement and that given in the summation notation. The formulas for the sum of consecutive integers, squares of consecutive integers, etc., were derived and the results were "checked" by the students as part of their laboratory assignment.

After the initial four lectures the students were assigned weekly laboratory projects. These projects were introduced by material prepared and written by the instructor. In addition to the description of the problem, further information about FORTRAN was included in the discussion of the project. Some of the topics introduced by this method were: more intricate formats, subscripted variables, and scientific subroutines. In this manner the students learned and developed better and more sophisticated programming techniques.

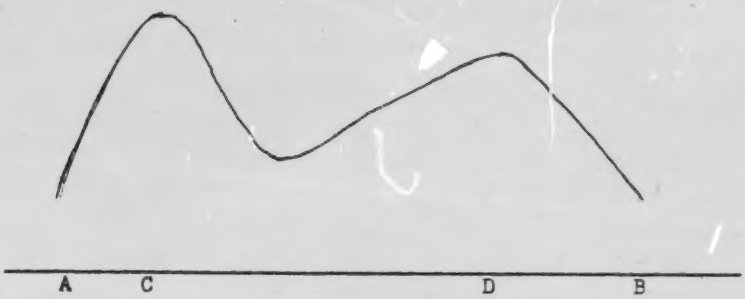
The actual work in the calculus began with a discussion of limits, continuity and the definition of the derivative. The computer problems from this section were aimed at attaining a better understanding of these concepts. Generally speaking, students have trouble with the epsilon-delta definition of a limit when it is presented in class, but after working with an algorithm to find all values of x "close" to 2 such that $x + 7$ is within .005 of 11, he has an understanding of what the definition is saying. This process can be repeated with other functions and values. The functions were deliberately chosen to be fairly routine. Thus the student can concentrate on the algorithm and not be distracted by an unnecessarily complicated function statement. It should also be noted that the student has a better understanding of the algorithm if he is able to actually check his output.

Attention was also given to the sequential definition of a limit. The students examined the values of the difference quotient of a given function for various terms of a convergent sequence. One of the problems was to evaluate the difference quotient sin h/h for $h = 2^{-N}$ $(N = 0,...,10)$. The result of this approximation provided the basis for future classroom discussion.

Another useful algorithm for use in this section is Newton's algorithm for approximating the square root of a positive number.

After the class had discussed the rules for symbolic differentiation, they were given an assignment on symbolic differentiation. This assignment utilized a program developed as part of an independent study project by a senior mathematics major at Gettysburg. Having the computer execute the symbolic differentiation did not really require any programming skill on the part of the calculus student, but it did emphasize the algorithmic nature of symbolic differentiation. Class discussion of the program required a renewed discussion of the sum, product, quotient and chain rules.

Optimization problems provide an excellent opportunity for stressing and appreciating the difference between computer science and mathematics. They also point out the limitation of each discipline. The algorithm for locating an extremum was the method of successive bisection of the interval and choosing either the right hand endpoint, the left hand endpoint, or the midpoint of the interval. The choice is dependent upon the value of the function at each point. A new interval is chosen and the process is repeated. This method is a brute force search and, in fact if the graph of the function is not sufficiently well behaved, the algorithm will not locate the correct extremum. For example, on a graph such as:



the algorithm will locate the local maximum at D and not the absolute maximum at C. This difficulty can be overcome by several methods, but at the expense of additional computer time. On the other hand, it is not necessary to make any differentiability assumptions about the function in question.

The development of the integral calculus involved several programs on the estimation of area, estimation of the total power consumption by a manufacturing company and the evaluation of upper and lower sums for monotonic functions on bounded intervals. These numerical calculations usually are, at best, tedious and time consuming in the usual classroom situation. Consequently, we usually avoid much of the numerical work leading up to the Fundamental Theorem of Calculus. This all too often results in the unfortunate situation that many students do not understand the distinction between an antiderivative and a definite integral. By programming the algorithm for either the Riemann sum or the upper and lower sums for a definite integral, the student not only attains a better understanding of the definition and properties of the definite integral, but also the meaning and power of the Fundamental Theorem of Calculus. This in itself makes the project an extremely worthwhile endeavor.

The final project of the first semester was the development of a method for approximating sine and cosine by polynomials. This algorithm is an especially useful one for reviewing the semester's work. First, we recalled from an earlier program that $\lim_{x\to 0} \sin h/h = 1$ (a fact we verified in class) and were thus able to find the derivative of sin x and cos x. The Fundamental Theorem of Calculus then gave us:

$$\int_0^x \sin t\,dt = 1 - \cos x \quad \text{and} \quad \int_0^x \cos t\,dt = \sin x.$$

Then, using the inequality $\cos x \leq 1$ and the properties of the definite integral, we have:

$$\sin x \approx x - \frac{1}{3!}x^3 + \dots + (-1)^{n+1}\frac{1}{(2n-1)!}x^{2n-1}$$
$$\cos x \approx 1 - \frac{1}{2!}x^2 + \dots + (-1)^n\frac{1}{(2n!)}x^{2n}$$

This process also yields the error estimates for these approximations. The students were asked to write a program for the above estimates: choose a value for x between 0 and π; determine the number of terms of the approximation which are necessary for four decimal place accuracy; and finally to estimate the value of sin x. The students were, in most cases, astounded at how few terms of the polynomial were required.

During the second semester we continued our coverage of calculus of a single variable with the examination of other transcendental functions. When we develop the formula

$$\int_1^x \frac{1}{t}\,dt = \ln x \qquad \text{for } x > 0$$

the student usually nods knowingly and feels that all his problems are solved. In reality, we have merely replaced one symbol with another symbol. By writing

$$\int_5^{13} \frac{1}{t}\,dt = \ln \frac{13}{5}$$

the student has no more appreciation for the numerical value of the integral, than he did before he solved the problem. In order to appreciate the meaning of this symbol he must refer to a table of values for the natural logarithm. The generation of such a table is not entirely in the spirit of the course but was felt to be a necessary project. This project does have the one redeeming feature that it reinforces the definition of the definite integral.

The introduction of the exponential function and its properties led to a discussion of elementary differential equations. In the course of the computer project we introduce Euler's method for approximating the solution of the equation. In order to make the results obtained by this method more meaningful, the student is introduced to the use of the plotter and his solution is displayed graphically.

The course concluded with a discussion of infinite series, Taylor series and power series. One of the problems considered in this section was, once again, the estimation of functions by polynomials. For example, by means of a change of variable we have:

$$\ln(1+x) = \int_0^x \frac{1}{1+t}\,dt, \qquad x > -1.$$

Noticing that

$$\frac{1}{1+t} = 1 - t + t + \dots + (-1)^{n-1} t^{n-1} + \frac{(-1)^n t^n}{1+t}$$

we integrate term by term and obtain an alternating series. Thus, for $0 < x < 1$:

$$\ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} + \dots + (-1)^{n-1}\frac{x^n}{n}$$

with an error no greater than $\frac{x^{n+1}}{n+1}$. Other problems of this type were also considered. Henriksen (2) has a particularly interesting example which illustrates the superiority of this type of an estimate for

$$\int_0^{1/3} \frac{dx}{1+x^3}$$

to the usual closed form solution of the integral.

Summary: The course described above is in no way a unique program for instituting the use of the computer in a calculus course. However, it does illustrate that it is possible to take such a step at a small college with limited facilities and a small mathematics department. One big factor contributing to the success of such a program is the availability of the computer facility to the students. Under normal operating conditions the student can expect a maximum turn around time of fifteen minutes. The computer center is available to the students from 8AM to 11PM seven days a week (these hours are slightly modified on Sunday).

One of the disadvantages of such a course is the heavy demand which is made upon the instructor's time. The lack of an adequate laboratory manual places a heavy demand for the coordinator of the course to carefully write mimeographed assignments. These assignments must not only include a discussion of the problems but also a development of any new programming techniques required by the assignments. There is simply not enough time available in the regular class to discuss the calculus and the programming of the computer assignment. This problem is slightly alleviated by the scheduling of a regular laboratory period. Also, one can safely assume that within the near future there will be a wide variety of laboratory manuals from which to choose.

A pleasant side effect of the program is the rapport which was established not only among the calculus students, but also, between the freshman calculus students and upperclassmen not in the calculus course. Many of the students who had trouble with the programming began consulting with their fellow students and the upperclassmen. The programming questions proved to be natural icebreakers.

Generally speaking, the students who finished the course were enthusiastic about their experience and also seemed to have a better feeling for the calculus. They had actually worked with the definitions and theorems of the calculus and seen the beauty of the optimization methods of the calculus, the power of the Fundamental Theorem, the meaning of convergence and divergence of infinite series and the value of learning numerical techniques in the calculus.

REFERENCES

1. L. Bers; Calculus; Holt, Rinehart and Winston; New York, N. Y. (1969).

2. M. Henriksen; "Calculus and the Computer: A Conservative Approach"; Proc. of a Conference on Computers in the Undergraduate Curricula; Univ. of Iowa; Iowa City, Iowa (1970), pp. 4.12-4.16.

3. G. J. Porter; "The Computer as a Teaching Aid in Mathematics"; Proc. of a Conference on Computers in the Undergraduate Curricula; Univ. of Iowa; Iowa City, Iowa (1970), pp. 4.17-4.19.

4. G. J. Porter; "A Computer Assisted Approach to Integrals via Jordan Content"; The American Mathematical Monthly; v. 77 (1970); p. 999-1001.

5. J. T. Schwartz; "Prospects of Computer Science"; The Mathematical Sciences (A Collection of Essays); M.I.T. Press; Cambridge, Mass.; (1969); pp. 261-271.

6. W. Stenberg, R. J. Walker, et al; Calculus, A Computer Oriented Presentation, parts 1-5; The Center for Research in College Instruction in Science and Mathematics; Florida State University; Talahassee, Florida (1968).

7. Calculus with Computers; CUPM Newsletter No. 4 (1969); CUPM Central Office; PO 1024, Berkeley, California.

THE COMPUTER AND THE CALCULUS

Theodore J. Benac
U. S. Naval Academy
Annapolis, Maryland

Technology has become a very important factor in modern educational practices. Up to a few years ago educational technology was restricted to audio-visual devices. Recently the computer has been added to the list. In certain areas, particularly in the physical and social sciences, its use is widespread and significant changes in course content have resulted. However, up to now its impact on the basic mathematical course has been minimal.

The formal training in mathematics of most students stops with the calculus. Due to the profound impact of the computer on applications of mathematics it would appear highly desirable to have the computer play a supportive role in the calculus. One must face the fact, however, that the syllabus of the traditional calculus course is already over-crowded. Consequently the inclusion of the computer cannot be justified unless it makes a positive contribution to the learning process.

How can the computer be involved in the calculus? One way is to use the computer merely as a calculating device. This can be accomplished effectively by teaching the calculus in the traditional manner based on a standard textbook and assigning problems to be solved by the computer. There are many places in the calculus where such assignments are appropriate: for example, in finding the roots of an equation by Newton's method and in evaluating definite integrals by Simpson's rule. The advantage of this approach is its obvious flexibility. Unfortunately, experience indicates that such computer involvement has little effect on increasing the student's understanding of the concepts under consideration.

Another way is to use the problem-solving capability of the computer to motivate concepts and proofs of theorems. Traditionally we have had recourse to elementary geometrical and physical ideas for this purpose. For example, by drawing curves we can easily justify the statements of important theorems such as the Mean Value Theorem, the positiveness of the derivative of an increasing function, the Fundamental Theorem of Calculus. Our experience with selected students over a two year period indicates that the computer can play a similar role. Furthermore, if the calculus is oriented to the computer this objective can be attained without an essential loss in material covered.

There are two serious problems which must be faced in setting up a computer-oriented calculus course. The first is concerned with the scarcity of textual material. Except for notes on isolated topics there is at present only one textbook suitable for general use. This is the CRICISAM text sponsored by the National Science Foundation with Professor Stenberg of the University of Minnesota and Professor Walker of Cornell University as principal authors.

The second is a staffing problem. At the present there are few instructors with classroom experience with computer-oriented calculus courses. Furthermore, most mathematics instructors are not only apathetic but downright antagonistic to the philosophy of such a course. Now experience shows that one does not have to be an expert on the computer or in numerical analysis to be effective in teaching a computer-oriented calculus course. However, it does require enthusiasm for the program. Furthermore, an instructor must be willing to forego the time-honored organization of the calculus and must detach himself from the 'plug and chug' syndrome. The latter is most important, for experience shows that if the 'discovery' and algorithmic approaches are used that the computer can serve as a most effective learning aid.

At the U. S. Naval Academy we have been giving a computer-oriented calculus course for the past two years based on CRICISAM. Last year we operated with 40 students, this year with 60. They were selected from volunteers for the program according to their pre-Academy record and CEEB scores. Though these groups were above average in ability, no attempt was made to conduct the course at an honors level. Actually, the pace set for the course was such that a student's workload was about the same as it would have been if he had enrolled in the regular calculus sequence.

How is the CRICISAM program implemented at the Naval Academy? First, programming is a part of the course. After experimenting with FORTRAN and batch-processing, our experience indicated that BASIC with time-sharing was much more effective and completely satisfactory for the requirements of the calculus. We have found four class hours spent exclusively on programming and one class hour on the computer as adequate. Mathematical content was included in these class hours by orienting examples and exercises towards calculations involving series, such as $\Sigma \frac{1}{n}$, $\Sigma (-1)^n \frac{x^{2n-1}}{(2n-1)^2}$. At the end of this exposure the students were able to write programs with loops and decision statements and had developed some skill in handling the sigma notation and indexing.

Second, epsilonics plays an important role in the course. This type of thinking is the natural follow-through to the decision making statements which are frequently used to stop the execution of a program. Though no attempt was made to develop skill in handling epsilonics abstractly, the involvement was intensive enough to warrant the appearance of the following type of questions on examinations: "Show that the function $f(x) = 2x^2 - x$ is uniformly continuous on $[1,3]$ by the ε-δ method," also "show that the sequence $(2n+1)/n$ has the limit 2 by the ε-N method."

Third, the algorithmic approach is used whenever possible. It has long been recognized that numerical algorithms are effective in motivating the concept of definite integral and this is naturally extensively exploited in this course. But it is also true that many other parts of the calculus lend themselves to an algorithmic treatment. One such example is the problem of determining the best linear approximation to a continuous function at a point. To illustrate how the computer and the calculus can interact three examples will be given. The material discussed is included in the computer-oriented calculus course taught at the Naval Academy.

Example 1. Immediately following the lessons on programming the following problem is considered. Its objective is to develop the concept of convergence for sequences and the distinguishing property of the real number system. It incidentally shows the naturalness of the ε-approach to the calculus.

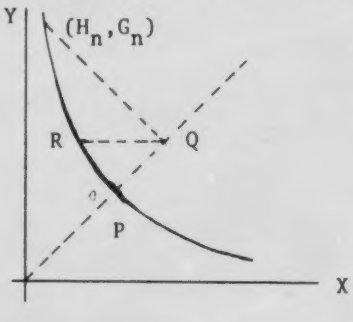
The student is assigned the following problem: write and execute a program to evaluate the terms of the sequences $\{G_n\}$ and $\{H_n\}$ which are defined recursively by

$$G_{n+1} = \tfrac{1}{2}(G_n + H_n), \quad H_{n+1} = A/G_{n+1},$$

where A is a given positive number. He is told to take as an initial value of G_1 an arbitrary positive number greater than A and to set $H_1 = A/G_1$. Further, he is directed to include a counter n in his program and a statement to stop the execution when $|H_n - G_n|$ becomes less than a given number E. Finally, his program should print the successive values of n, H_n, G_n. It is recommended that the student run the program for various values of E.

In this program the number E plays the role of a tolerance factor and it is associated with the term of least index passing the test of accuracy. One is then led to speculate whether the terms after the last one computed also pass the test of accuracy and whether the sequences (G_n) and (H_n) are zeroing in on some number L. With the introduction of the concept of an E-approximation to L one has thereby developed an appropriate background for a theoretical discussion of convergence.

What is the number L? A geometric approach is very instructive.



In the xy-plane consider the set of points with coordinates (H_n, G_n). The relation $H_n = A/G_n$ or $H_n G_n = A$ implies that these points all lie on the hyperbola $xy = A$. Let P be the point of intersection of this hyperbola with the line $y = x$ and (a,a) its coordinates. Then $a^2 = A$, so that $\sqrt{a} = A$. Let Q be the foot of the perpendicular dropped from (H_n, G_n) to the line $y = x$. Its coordinates are readily shown to be (G_{n+1}, G_{n+1}), since

$$\frac{G_{n+1} - G_n}{G_{n+1} - H_n} = \frac{\frac{1}{2}(G_n+H_n) - G_n}{\frac{1}{2}(G_n+H_n) - H_n} = \frac{-\frac{1}{2}G_n + \frac{1}{2}H_n}{\frac{1}{2}G_n - \frac{1}{2}H_n} = -1 .$$

Hence the point R which is the horizontal projection from Q to the hyperbola has coordinates (H_{n+1}, G_{n+1}). Geometrically it appears that the points (H_n, G_n) are approaching P, which would mean that $L = \sqrt{A}$.

In what sense are the sequences approaching $\sqrt{A}$? One first notes that

$$|G_{n+1} - H_{n+1}| \leq |\tfrac{1}{2}(G_n+H_n) - H_n| = \tfrac{1}{2}\,|G_n - H_n|.$$

Hence by recursion we get

$$|G_{n+1} - H_{n+1}| \leq \frac{1}{2^n}|G_1 - H_1| .$$

Now let N be chosen so that $\frac{1}{2^N}(G_1 - H_1) \leq E$. It is clear that

$$|G_{n+1} - H_{n+1}| \leq E$$

for all $n > N$.

Finally, an obvious geometrical argument yields the inequalities

$$H_n \leq H_{n+1} \leq \sqrt{A} \leq G_{n+1} \leq G_n$$

Hence

$$|H_n - \sqrt{A}| \leq |H_n - G_n| \leq E ,$$

which says that the sequence $\{H_n\}$ is an E-approximation to $\sqrt{A}$ for any positive number E; similarly for $\{G_n\}$.

The above discussion has established that the algorithm executed by the computer is a procedure for finding the square root of an arbitrary positive number. More important is the fact that it provides a significant motivation and formulation for the following two concepts which play a basic role in the development of the calculus. First, a sequence $\{a_n\}$ is said to converge to a number L if for every $\varepsilon > 0$ there exists a positive integer N such that for all $n > N$ the inequality $|a_n - L| \leq \varepsilon$ is valid. The number L is called the limit of the sequence $\{a_n\}$ and we write $\lim_{n\to\infty} a_n = L$. Second, let $\{a_n\}$ be an increasing sequence and $\{b_n\}$ a decreasing sequence such that the sequence $\{a_n - b_n\}$ converges to zero. The Completeness Axiom of the real number system states that there exists a real number L such that $\lim_{n\to\infty} a_n = L = \lim_{n\to\infty} b_n$.

The next two examples are given to illustrate that certain basic theorems of the calculus need not be considered as theorems whose proofs are beyond the scope of the freshman calculus course. There are two factors which make this possible. First, algorithmic proofs of these theorems can be given. Second, the consideration of these theorems takes place only after the student has had extensive experience writing programs based on the bisection method.

Example 2. This example is concerned with the Intermediate Value Theorem. Let f be a real-valued function, continuous on the closed interval $[a,b]$ and I a real number such that $f(a) \le I \le f(b)$. The student is assigned the problem to write and execute a program to find a number c in the interval $[a,b]$ such that $f(c)$ is a good approximation to I. He is directed to use a bisection algorithm.

Starting with $L_1 = a$ and $R_1 = b$ the quantity $M_1 = (L_1+R_1)/2$ is computed, thereby forming two intervals (L_1,M_1) and (M_1,R_1). One of these intervals is discarded by the statement:

If $f(M_1) < I$, then let $L_2 = M_1$, $R_2 = R_1$; otherwise, let $L_2 = L_1$, $R_2 = M_1$.

The procedure is repeated using the interval (L_2,R_2) instead of (L_1,R_1) to yield a new interval (L_3,R_3), etc. The program should print the value of M_n and the execution should stop when $|L_n-R_n|$ is less than some prescribed tolerance factor ε.

Clearly the algorithm yields an increasing sequence $\{L_n\}$ and a decreasing sequence $\{R_n\}$. Further

$$|L_{n+1}-R_{n+1}| = \tfrac{1}{2}\,|L_n-R_n| = \dots = \frac{1}{2^n}\,|L_1-R_1|$$

implies that the sequence $\{L_n-R_n\}$ converges to zero. Hence by the Completeness Axiom there exists a real number c such that

$$\lim_{n\to\infty} L_n = c = \lim_{n\to\infty} R_n .$$

The inequalities $L_n \le c \le R_n$ and $L_n \le M_n \le R_n$ clearly state that the value of M_n obtained at the termination of the program is an ε-approximation to c.

The proof of the Intermediate Value Theorem can now be easily completed by using the continuity property of the function f. First we note that the algorithm yields the inequalities

$$f(L_n) \le I \le f(R_n) ;$$

second, that the definition of continuity states

$$\lim_{n\to\infty} f(L_n) = f(\lim_{n\to\infty} L_n).$$

A simple argument establishes that $f(c) = I$.

It should be noted that the proof of the Intermediate Value Theorem is based on a construction which can be computerized. The next example is concerned with a case in which, though the algorithmic proof is not executable, it is a natural outgrowth of a numerical algorithm. Experience has shown that the algorithmically oriented student has no difficulty coping with it.

Example 3. This example deals with the Maximum Property of a continuous function. The student is given a real-valued function f continuous on the closed interval (a,b) with only one local maximum in the interval and is directed to locate the maximum point by a search method. Among the various possibilities the following bisection procedure is suggested.

Starting with $L_1 = a$, $R_1 = b$, $H = (b-a)/2$ the quantities $H_1 = H/2$ and $M_1 = (L_1+R_1)/2$ are computed and the points C_1, D_1 are located by the $C_1 = M_1 - H_1$, $D_1 = M_1 + H_1$. The objective of the program is to replace the interval (L_1,R_1) by an interval half its length with some assurance that the discarded part of (L_1,R_1) will not contain the maximum point. A geometric argument indicates the appropriateness of the following statement:

If $f(c_1) \geq f(M_1)$, let $L_2 = L_1$, $R_2 = M_1$;

if $f(c_1) < f(M_1)$ and $f(D_1) \geq f(M_1)$ let $L_2 = M_1$, $R_2 = R_1$;

otherwise, let $L_2 = C_1$, $R_2 = D_1$

The procedure is repeated by computing $H_2 = H_1/2$, $M_2 = (L_2 + R_2)/2$ and locating C_2 and D_2 about M_2, etc. The program is executed until the change in the value of the function f for two successive values of M_n becomes less than some tolerance factor ϵ and the values of M_n and $f(M_n)$ printed.

Again, it is clear that the algorithm yields an increasing sequence $\{L_n\}$ and a decreasing sequence $\{R_n\}$ such that the sequence $\{L_n - R_n\}$ converges to zero, so that by the Completeness Axiom the sequences $\{L_n\}$ and $\{R_n\}$ converge to a real number m. Unfortunately, it does not follow that f(m) is a maximum value for the function f. Such a conclusion would require that f(m) had been tested against f(x) for all x in the interval $[a,b]$, not just those picked by the algorithm.

The proof of the Maximum Theorem is handled by adjusting the decision statement so that the above requirement is met. However there is no way of doing this by an executable statement. The following is satisfactory:

If there is an x in (L_n, M_n) such that $f(x) \geq f(y)$

for all y in (M_n, R_n), then let $L_{n+1} = L_n$, $R_{n+1} = M_n$;

otherwise, let $L_{n+1} = M_n$, $R_{n+1} = R_n$.

The number m that is thereby obtained can be shown to have the desired property. Thus, one notes that if $x \neq m$ is an arbitrary number in the interval $[a,b]$, then at some stage of the algorithm the two numbers m and x_1 must lie in different half-intervals and when this happens the algorithm will choose the half-interval I containing m, indicating that I contains a number x_2 with the property $f(x_2) \geq f(x_1)$. If $x \neq m$, one repeats the argument to acquire a new number x_3 such that $f(x_3) \geq f(x_2)$, etc. Clearly

$$\lim_{n \to \infty} x_n = m.$$

The continuity of f and $f(x_n) \geq f(x_1)$ yields $f(m) \geq f(x_1)$.

Before closing two observations based on our experience teaching a computer-oriented calculus course are significant. The first is that such a course is not for everyone. The only way some students can learn mathematics is by extensive drill and through the exhaustive treatment of each topic. The integration of the computer with the calculus does not permit such a modus operandi without a significant increase in the time allotted to the course. Our experience indicates that no serious involvement with the computer should be attempted in the calculus unless dealing with students who are interested in and have the ability to comprehend the 'why' of mathematics. With such students the educational practice of teaching by exhaustion is not necessary.

The second observation is that it is worthwhile to give the multivariate calculus a computer-oriented treatment. This past semester such a course was taught with the CRICISAM Calculus Course as a prerequisite. It was based on a standard text with computer supplementation. Included in the course were search techniques to investigate discontinuities and to find maximum points using grids as well as contour curves, an algorithmic approach to the line integral and the double integral, a discovery approach to determine the best linear approximation to a surface at a point. Furthermore, at this stage our students were interested in acquiring more sophisticated programming skills. This was exploited by introducing subscripting and subroutining. Their use was especially effective in evaluating double integrals by Simpson's rule. Our experience indicated that a significant improvement resulted both in the coverage and understanding of the elements of the multivariate calculus.

CALCULUS AND THE COMPUTER: A PARALLEL APPROACH

J. C. Mathews
Iowa State University
Ames, Iowa

During the past six or seven years the use of computers in the undergraduate mathematics curriculum has developed from a few widely scattered experimental programs at institutions such as Dartmouth, University of New Mexico, University of Utah, Oberlin College, and Florida State University to the present diverse activity at several hundred institutions. Some programs have just begun this year, others are past the experimental stage and are becoming an accepted and useful adjunct or addition to their undergraduate mathematics curricula, and still others are now fully accepted and utilized parts of environments in which mathematics is being learned.

The arguments that have been given in support of the use of computers in undergraduate mathematics are of several different kinds, depending upon such factors as

A. The kind of mathematics being considered; e.g., analysis, algebra, finite mathematics, numerical analysis, or logic.

B. The relationship between the mathematical and the computing activities; e.g.,
    1. Rather completely intertwined, where the computing activities are used within the mathematics course at the time and in the amount needed to motivate or to illustrate a numeric, computational approach to a mathematical idea.
    2. Parallel, with their content and schedules rather carefully coordinated.
    3. Supplemental, where perhaps the major emphasis is on the computer and programming rather than the mathematical content.

C. The acceptance of the idea that many -- perhaps most -- students should learn to use the data handling and computational power of modern computers as early as possible, so that they are aware of both the uses and limitations of high speed, large capacity computers in their subsequent work.

In analysis, particularly in elementary calculus at those points where a new limiting process is being introduced (e.g., sequential convergence, the definite integral, infinite series), an argument can be made that well chosen computing activities can strongly motivate and illustrate both the "gut feel" and the details of the limit process. The analysis of the error and the use of "Stop Rules" are effective means for leading the student to a deeper understanding of a limit process and the definitions ustd to describe it. Often, the student's involvement with a program and its structure forces a more thorough consideration of the mechanics of the limit process than is otherwise attained. Apart from the limit process as such, there are several topics in calculus that are much more realistically and ably discussed with the help of an accessible and reasonably large computer. Such topics as Newton's method, finding the maximum of a continuous function on a closed interval, the trapezoidal or Simpson's rule, and function approximation with Taylor's formula fall into this category. Although the error analysis for various algorithms is often more intricate than can be handled in a freshman course, enough can be said about it and enough awareness of its necessity arises out of the programming of various algorithms to be of considerable value to the students. An interesting discussion of using computers in calculus is given by M. Henriksen in the Proceedings of a Conference on Computers in the Undergraduate Curricula, 1970, The University of Iowa. The papers by Robert Kuller and G. Albert Higgins in this Proceedings also contain interesting ideas for using the computer in a practical and productive way in a standard calculus course. An excellent discussion of the pitfalls in numerical calculation, a topic that should accompany any error analysis, is given by George Forsythe in the November, 1970 issue of the American Mathematical Monthly.

In what follows I wish to review our experience at Iowa State University with the use of computers in our freshman calculus course and then to describe our plans for the coming year.

In 1969-1970 Professor Eugene Steiner and I taught elementary calculus in an environment including student access to an IBM 360/65 computer via two remote IBM 2741 typewriter consoles. We used the CRICISAM text for the course. A combination of the nature of the material presented, the fact that both of us were involved in the teaching, and our differing views on education, led us to divide the week into three parts. For the first three days one of us introduced the new topics to be studied, exploiting the algorithmic approach whenever possible. There was a considerable amount of variation among the topics in terms of how naturally they could be presented from an algorithmic viewpoint. In these three days the students were assigned homework in an amount and of a kind roughly similar to an ordinary calculus class. Of the remaining two days in a typical week, one of them was devoted to drill problems. A small part of this fifty minute period ran on the standard format wherein the instructor answers questions raised by the students. In the remaining portion of the period a work-sheet was handed out, covering the topics presented in the preceding three days. The students were encouraged to work together and the instructor answered any questions that were asked, on an individual basis. This proved to be a very efficient and interesting way of involving almost all of the class in some kind of mathematical activity. On the last day of the week, a problem to be solved on the computer was discussed in detail. The problem usually was directly related to one of the topics discussed during the week. The students were expected to find a method of solving the problem, to program this solution, and to run the program on their owt time during the following week.

We began the year with 29 students and ended the year with 17 students, a rate of attrition that we judged to be not significantly different from that of the normal calculus class. We found it difficult to obtain quantifiable judgements of the value of our computer-oriented approach to calculus because, despite the basic similarity in subject matter with the standard course, the order of topics was different, the points of special emphasis were different, the pace was much more varied, and some topics were deleted to make room for the algorithmic and computer-related topics. We therefore have only our own judgements that the results obtained with the experimental, computer-calculus section were very encouraging. It may be true that in manipulative ability--which decays rapidly in most students--the students in the regular section would have proved more adept. On the other hand, we feel that our students have a better understanding of some of the basic ideas of calculus, are much more aware of and proficient in the computational side of calculus, and have gained attitudes toward and knowledge of computers that will be useful to them later.

During the year of our experimental computer calculus course we applied for and subsequently received an NSF grant to buy seven Model 35 Teletypes equipped with Hewlett-Packard Optical Mark Readers to partially equip a Mathematics Laboratory. The installation of these machines will not be complete until this summer because of delays in delivery and necessary local modifications in this equipment. We anticipate that the first routine use of the laboratory will occur this fall.

As at many universities with a large engineering college, our large service courses such as elementary calculus and differential equations are high topic-density, rapidly moving courses. As Henriksen has said

> "...any effective approach to integrating numerical methods into the calculus has to be a conservative one...only a limited amount of new material would be tolerated, and it would have to blend naturally into the standard topics of calculus. In particular, no large amount of the student's time could be occupied with programming, and one could not rely on the ability or willingness of the typical calculus instructor to teach programming or supervise programming activity."

For such reasons as these we have decided to use our computing facilities in a parallel mode, with the students electing to take in conjunction with a more or less standard mathematics course a one credit laboratory course that is roughly parallel in content and schedule to the standard course. At the outset we hope to have such laboratory courses available in elementary calculus, differential equations, and finite mathematics.

We hope that eventually the "standard courses" come to reflect the existence of convenient and powerful computational facilities and what appears to be a rapidly developing and broad-based opinion in the mathematics community favorable to the use of computers in the mathematics curriculum. At the present time, however, particularly in large service courses, the innovative work will be done in the parallel laboratory course and not in the standard course.

For the present, then, we have assumed that most of the teaching in our elementary calculus will be rather traditional. With a population of 2000 students in elementary calculus and classes of 50 students each, this amounts to 25 to 35 instructors. It is difficult, without a textbook to support a common philosophy among a group this large (and diverse), to achieve much uniformity, sustainable enthusiasm, and interest in teaching from an algorithmic viewpoint, wherein emphasis is placed on the computational side of calculus and there is some risk of becoming involved in some aspects of programming.

The one credit laboratory course paralleling elementary calculus is designed to help the student learn (1) a computer language and programming, both as minimal as possible, (2) what an algorithm is, concentrating on those that arise in the concurrent calculus course, (3) the need for and some techniques of error analysis, and (4) something of the need and practice of numerical methods in calculus. We hope to achieve these objectives through a series of specific problems rather than in tidy lectures, through the frustrations of faulty logic, bad programming, and computer breakdowns, and by encouraging the students to help each other and to freely consult (and copy from, if they wish) other students, the undergraduate assistants, or some faculty member. We plan to use undergraduate assistants in the computer room. Their primary responsibilities are to help with the mechanics of entering programs into the computer and to help the students over their difficulties with programming. A faculty member will be available as a consultant on the mathematical questions that will, if the problems are set well, arise.

This leads to the last major topic I wish to discuss, the one, in my opinion, that is most important to the success of the parallel laboratory course. I am referring to the collection of problems that will bear the structure and intent of the course, and which will help in giving the student who has completed the calculus course with laboratory a view of calculus that takes into account the impact of the computer on science and technology. It is difficult to describe the kind of problems that will accomplish these ends. I have at times attempted to list some characteristics of the kind of problems that I have found suitable, but the results have been little more than vague, non-operational, and largely futile exercises. I am therefore merely giving a few examples of the problems that have appeared to be most useful to the students.

1. a. For $n = 4, 8, 16, \ldots, 128$, find the lower and upper estimates L_n and U_n of the area A of a circle of radius 1 by using the n-sided inscribed and circumscribed polygons.

   b. Show that if you estimate A by the average $(L_n + U_n)/2$ of the lower and upper estimates, the error is less than $(U_n - L_n)/2$.

   c. Based on the data you have, give an estimate of A and a bound on the error of your estimate.

2. a. Write a program that gives the maximum value and a maximum point of any quadratic function

$$f(x) = ax^2 + bx + c \qquad u \leq x \leq v$$

   where a, b, and c are real numbers, $a \neq 0$, and u and v are real numbers such that $u < v$.

   b. Use your program to find the maximum value and a maximum point of each of the following functions:

      i. --(a quadratic that takes on its maximum at an interior point)

      ii. --(a quadratic that takes on its maximum at an end point)

3. For $t > 0$ the relation

$$Vt + V^3 = -1$$

   defines V as a function of t.

   a. Use Newton's method to calculate V at $t = 3$ and $t = 3.1$.

   b. Calculate V at $t = 3.1$ using the differential of V at $t = 3$.

4. The equation $1 - y + \sin(xy) = 0$, where $0 < x \leq 1$, implicitly defines a function f which is continuous on $[0,1]$ and differentiable on $(0,1)$.

   a. Write and run a program that computes $f(.1),\ldots,f(1.0)$ with sufficient accuracy that

   $$|1 - f(.1) + \sin(.1f(.1))| \leq 10^{-3}$$
   $$|1 - f(.2) + \sin(.2f(.2))| \leq 10^{-3}$$

   and so on.

   b. Using the data from part a make a careful graph of f and estimate the maximum value of f and a maximum point.

   c. Find the maximum of f and a maximum point by non-numerical methods.

These four problems hardly begin to cover the topics that are worth consideration. I have given them to illustrate the type of problem that I have found to be sufficiently challenging, interesting, and useful to the students in elementary calculus.

It is very difficult to make reliable estimates of costs per student per year for the type of computer usage I have discussed. Academic Deans, however, do enjoy having such information. The estimate of direct costs for 1971-72 that I recently submitted amounted to about $36 per student per year.

COMPUTER (PROGRAMMABLE CALCULATOR) SUPPLEMENTED ORDINARY DIFFERENTIAL EQUATIONS

Gene A. Kemper
University of North Dakota
Grand Forks, North Dakota 58201

INTRODUCTION

During the 1970-71 academic year the usual undergraduate ordinary differential equations course at the University of North Dakota was replaced by computer supplemented ODE material that was taught without a text. The primary objective of the course was to give the students insight into theory and applications. Less emphasis was placed on classical methods of solution. Such methods are available in other courses.

Although an IBM 360/30 was available for student use the students used an Olivetti-Underwood Programma 101 programmable calculator, available for under \$4000. (Figure 1) The 101 has eight storage registers each having a capacity of 22 digits, plus decimal point and sign. Three of these registers are operating registers and take part in the arithmetic operations. Each of the remaining five registers can be split into two registers each with a capacity of 11 digits, plus decimal point and sign. Three of the registers (or six split registers) can also be used for instruction storage. Use of these three storage registers for instruction storage increases the number of instructions allowed from 48 to 120. The 101 has branching capability and built in square root and absolute value operations. The precision of the arithmetic operations can be controlled by a decimal setting allowing the operations to be performed with from zero to fifteen decimal place accuracy.

Two hours of classroom time was used for programming instruction. The 101 was used both in and out of the classroom. Several of the assigned programs were stored on magnetic cards. Three times during the semester each student was scheduled for an oral examination which consisted of more in depth investigation of certain assigned problems. The students were expected to utilize the 101 during these exams. Written exams were also given.

THE COURSE

Detailed material regarding scalar first order linear ODE's will be shown here. Other topics will be discussed in less detail.

A. A Discrete Model: Difference equations were used to develop a mathematical model of population growth. In particular, if N(t) represents the size of a population at time t, r_b represents the birth rate and r_d represents the death rate then assume the "average rate of change" in the population in a time increment Δt is given by

$$\frac{N(t+\Delta t)-N(t)}{\Delta t} = (r_b - r_d)N(t), \qquad (1)$$

or

$$\Delta N(t) = (r_b - r_d)N(t)\Delta t$$

where $\Delta N(t) = N(t+\Delta t) - N(t)$. Letting $N_k = N(k\Delta t)$ there follows

$$N_{k+1} = N_k + (r_b - r_d)N_k\Delta t, \; k = 0,1,2,\ldots \qquad (2)$$

Utilizing the computer in class we investigated the situation where

Case I: $r_b = \beta, r_d = \delta$, β and δ constants.

Then (2) becomes

$$N_{k+1} = \left[1 + (\beta - \delta)\Delta t\right] N_k. \qquad (3)$$

Equation (3) was studied using $\beta = 1.5$, $\delta = 1, N_0 = 2$ and $h = 0.1, 0.2$. The resulting program and output is given in Figure 2 with the corresponding graph in Figure 3. The precision setting corresponds with the number of decimal places in the output.

The class was divided into three groups each of which investigated one of the following cases:

Case II: $r_b = \beta N_k,\ r_d = \delta$

$$N_{k+1} = (1 - \delta\Delta t)N_k + \beta\Delta t N_k^2 \tag{4}$$

Case III: $r_b = \beta,\ r_d = \delta N_k$

$$N_{k+1} = (1 + \beta\Delta t)N_k - \delta\Delta t N_k^2 \tag{5}$$

Case IV: $r_b = \beta N_k,\ r_d = \delta N_k$

$$N_{k+1} = N_k + (\beta - \delta)\Delta t N_k^2 \tag{6}$$

In all cases, β and δ are constants. In particular, so that comparisons could be made, all students used $\beta = 1.5$, $\delta = 1$, $N_0 = 2$, $h = .1$ and $.2$. The programs, output and corresponding graphs are shown in Figures 4 through 9.

When all these investigations were complete, further investigations were done in class. The quantity N_0 was replaced by αN_0, $\alpha = 1/2, 2, 3$. Three of the conclusions arrived at were:

(a) When N_0 was replaced by αN_0 the N_k's resulting from (3) were a factor of α times the N_k's obtained from (3) using N_0. This did not happen when (4), (5) and (6) were used. Apparently, according to the students, this was due to the nonlinear terms in (4), (5) and (6).

(b) For $N_0 = \frac{1}{2}, 1, 2, 3, 4$ the solutions of (5) tended to 1.5 thus providing an introduction to stability.

(c) In some sense the solutions are continuous with respect to the initial value N_0.

B. First Order Linear ODE's: From their investigations and the mathematical model the students hypothesized that the smaller the time increment Δt the closer their results would be to the actual population growth based on the stated assumption. This led to the consideration of the limiting case, $\Delta t \to 0$. From (1), as $\Delta t \to 0$, there follows

$$\frac{dN(t)}{dt} = (r_b - r_d)N(t), \tag{7}$$

the first ODE encountered in class. For the four cases (7) becomes:

Case I: $\dot{N} = (\beta - \delta)N$ (8)

Case II: $\dot{N} = (\beta N - \delta)N$ (9)

Case III: $\dot{N} = (\beta - \delta N)N$ (10)

Case IV: $\dot{N} = (\beta - \delta)N^2.$ (11)

Because the output in Figure 3 appeared to be of exponential form, (8) was assumed to possess a solution of the form $\alpha e^{\lambda t}$, α and λ constants. The students noted when $\alpha e^{\lambda t}$ was substituted into (8) that α cancelled out, provided $\alpha \neq 0$. This further reinforced linearity and led to the concept of a trivial solution. Furthermore, it was necessary to choose α such that $N(0) = N_o$, i.e., an encounter was made with ODE initial value problems.

The method just described did not work on (9), (10) or (11) so the method of "variables separable" was introduced. The solutions of (8), (9), (10), and (11) satisfying $N(0) = N_o$ are

$$N(t) = N_o e^{(\beta - \delta)t}, \tag{12}$$

$$N(t) = \frac{\delta}{\beta - \left[(\beta N_o - \delta)/N_o\right] e^{\delta t}}, \tag{13}$$

$$N(t) = \frac{\left[\beta N_o/(\beta - \delta N_o)\right] e^{\beta t}}{1 + \left[\delta N_o/(\beta - \delta N_o)\right] e^{\beta t}} \tag{14}$$

and

$$N(t) = \frac{N_o}{1 - (\beta - \delta)N_o t} \qquad (15)$$

respectively. For $\beta = 1.5$, $\delta = 1$ and $N_o = 2$ these solutions are illustrated in Figures 3, 5, 7 and 9, respectively. By examining these graphs and the output the students gained insight concerning step size t versus accuracy and precision of arithmetic operations versus accuracy. Furthermore, Case IV indicates how the solution of the difference equation (6) "neglects" the discontinuity in the solution (15) of (11) but does give an indication of this discontinuity. The stability in Case III was further investigated using a direction field.

The Euler-Cauchy method was discussed geometrically and analytically. Noting that

$$N(t + \Delta t) = N(t) + \dot{N}(t)\Delta t + \ddot{N}(\xi)\frac{(\Delta t)^2}{2} \qquad (16)$$

the truncation error is $\phi((\Delta t)^2)$. The geometrical discussion explained why the difference equation solution was below (or above) the solution of the corresponding ODE.

During the time these investigations were being conducted class time was used to develop theory and solution techniques of first order linear ODE's. The following is a sample assignment problem:

Example 1. A parachutist jumps from an airplane. The air resistance is proportional to the speed of descent. Assume the parachutist falls vertically, that his chute is open immediately when he jumps and that he falls from rest. If the weight of the chutist and his gear is w(pounds) and the constant of proportionality due to air resistance is show the mathematical model is

$$\frac{w}{g}\dot{v} = w - \beta v, \ \beta > 0$$
$$v(0) = 0,$$

where v is the velocity of descent measured positively downward and g is the acceleration due to gravity (32 ft/sec^2). Find the solution of the initial value problem. Sketch the graph of the solution. Find the limiting velocity v^∞ defined by

$$v^\infty = \lim_{t\to+\infty} v(t).$$

If x(t) represents the distance the chutist has fallen in time t seconds after the jump then since velocity is the time rate of change of position it follows that

$$\frac{dx(t)}{dt} = v(t) \text{ and } x(0) = 0$$

Find x(t).

Suppose β is to be determined using the relation $w = \beta v^\infty$ determined above. Furthermore, suppose n chutists with weights $w_1, w_2, \ldots w_n$ each jump and their corresponding limiting velocities are $v_1^\infty, v_2^\infty, \ldots, v_n^\infty$, respectively. Choose β so as to minimize the function ϕ given by

$$\phi(\beta) = \sum_{k=1}^{n} (w_k - \beta v_k^\infty)^2 .$$

The determination of β was to illustrate how least squares can be applied to experimental results to estimate a physical parameter.

C. Second Order ODE's. Second order ODE's were not introduced from difference equations. Rather, a spring-mass system and an electric circuit were considered both of which had a mathematical model of the form

$$\left.\begin{aligned} &\ddot{x} + \dot{a}_1 x + a_2 x = 0 \\ &x(0) = x_o, \ \dot{x}(0) = \dot{x}_o \end{aligned}\right\} \qquad (17)$$

An analogy between the mechanical and electrical system was discussed. Then, a numerical method for solving (17) was sought. To this end, using Taylor's series, it was noted that

$$\dot{x}(t) = \frac{x(t + h) - x(t)}{h} + 0(h), \qquad (18)$$

$$\dot{x}(t) = \frac{x(t + h) - x(t - h)}{2h} + 0(h^2) \qquad (19)$$

and

$$\ddot{x}(t) = \frac{x(t + h) - 2x(t) + x(t - h)}{h^2} + 0(h^2), \qquad (20)$$

where we now use $h = \Delta t$. The merits of using (19) and (20) instead of (18) and (20) were discussed. Substituting (19) and (20) in (17) eventually yields

$$(1 + \frac{a_1h}{2})x_{k+1} + (a_2h^2 - 2)x_k + (1 - \frac{a_1h}{2})x_{k-1} = 0, \quad k=1,2,3,\ldots \qquad (21)$$

To solve (21) we need x_o and x_1. $x_o = x(0)$ is known. To obtain x_1 Taylor's series was used in the form

$$x(h) = x(0) + \dot{x}(0) + \ddot{x}(0)\frac{h^2}{2} + \ldots$$

Now $x(0) = x_o$ and $\dot{x}(0) = \dot{x}_o$ are known. $\ddot{x}(0)$ can be obtained directly from (17). If desired, the values of higher order derivatives of x evaluated at $t = 0$ can be obtained by differentiating (17) and evaluating at $t = 0$. For example, retaining terms through h^3 yields

$$x_1 = (1 - \frac{a_2h^2}{2} + \frac{a_1a_2h^3}{6})x_o + (h - \frac{a_1h^2}{2} + \frac{(a_1^2 - a_2)h^3}{6})x_o . \qquad (22)$$

Using (22) to determine x_1 a solution of (21) can be generated. The students did this and compared the discrete solution to the solution of (17) as obtained from theory and techniques developed in class.

D. Two Point Boundary Value Problems: The method of the previous section ((21) and (22)) can be used to find an approximate solution of the two-point boundary value problem

$$\left.\begin{aligned} &\ddot{x} + a_1\dot{x} + a_2x = 0 \\ &x(0) = x_o,\ x(t_f) = x_f \end{aligned}\right\} \qquad (23)$$

The "shooting method" was employed. Two estimates for $\dot{x}(0)$ were used to obtain the errors between the corresponding solution and x_f. Linear interpolation was used to obtain the next estimate for $x(0)$. This process was repeated until a solution of the difference equation was obtained that was within a preassigned tolerance of x_f.

E. Systems. Systems of equations were introduced by the following pursuit problem.

Example 2: A rabbit, initially at $(x_o,0)$, runs along the positive x axis with constant speed S_r. A dog, initially at $(0,y_o)$, runs always toward the rabbit at a constant speed S_d. Determine the path of the dog.

A discrete model for this example was developed and investigated. Various speeds and initial positions of the dog and rabbit were considered.

Some two dimensional systems of first order autonomous systems were numerically investigated. The Euler-Cauchy method was utilized with the approximate solutions being compared to the actual solution. Some elementary properties of autonomous systems were discussed.

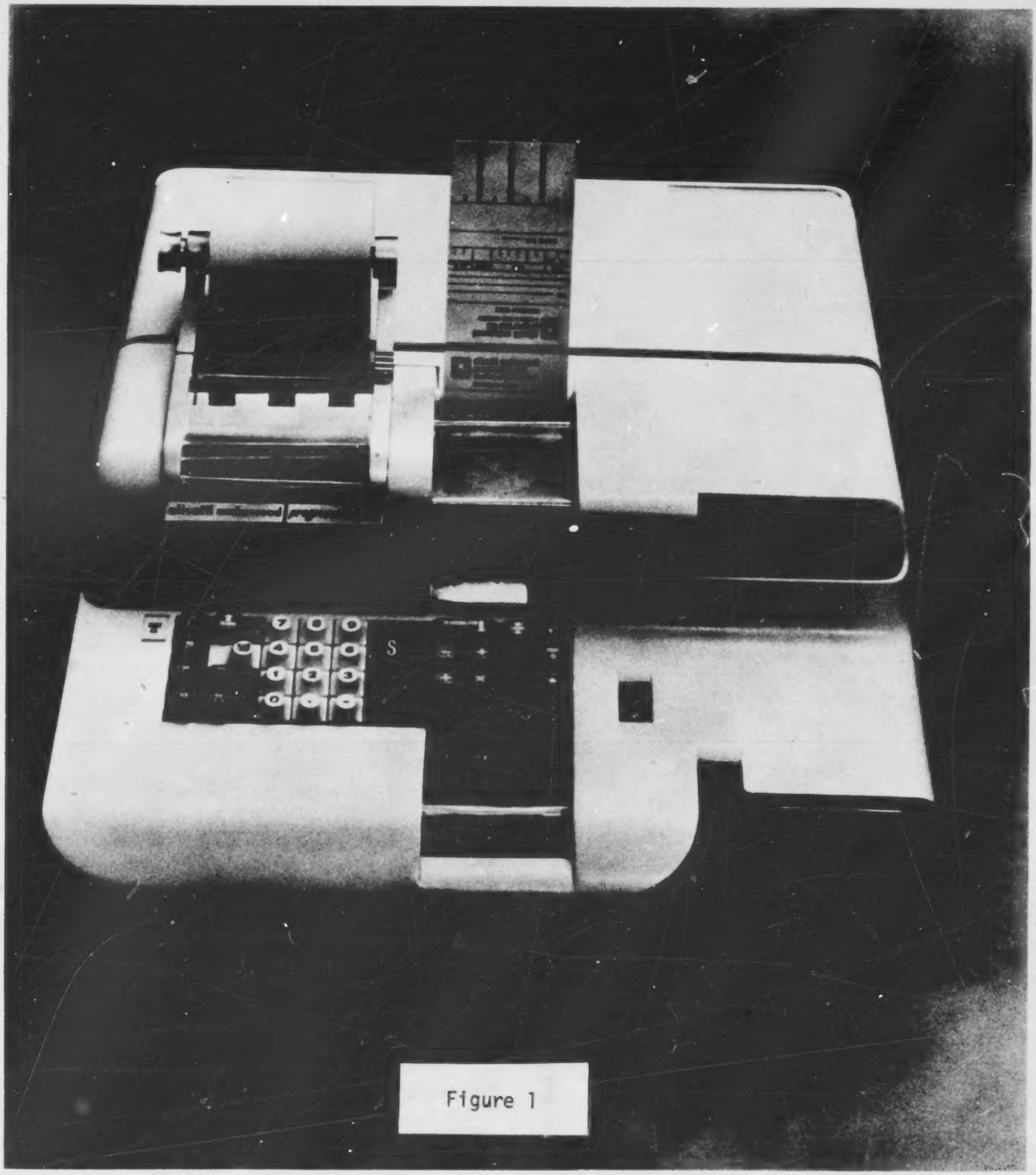


Figure 1

```
AV
MS
b ↑
AY
d *
MS
M ↓
MS
M −
MS
MY
b +
B ↕
B ◊
MS
c ↑
MS
C ↑
a W
B ↓
C X
A ◊
C ↕
d ↓
b +
d ↕
c ↓
d −
/W
MY
```

```
    1   S
  1•5   S
    1   S
  0•1   S
 1•05  B◊
   10   S
    2   S
 2•10  A◊
 2•20  A◊
 2•31  A◊
 2•42  A◊
 2•54  A◊
 2•66  A◊
 2•79  A◊
 2•92  A◊
 3•06  A◊
 3•21  A◊
```

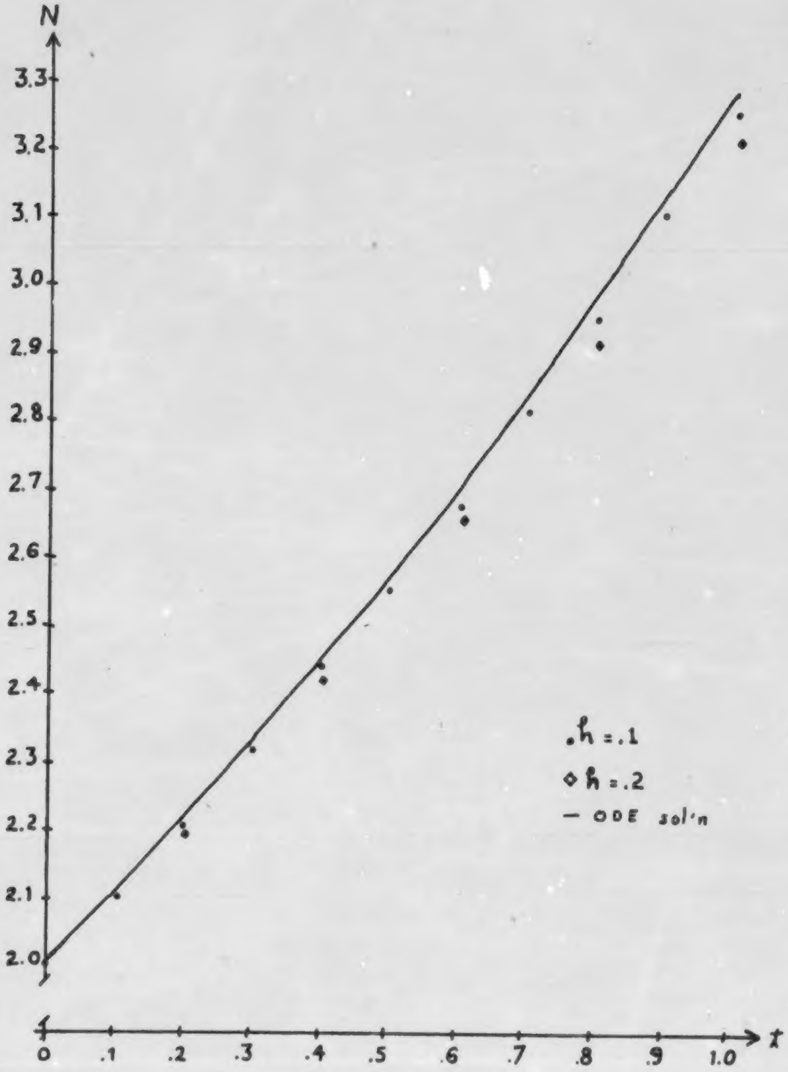
```
    1•5    S
      1    S
    0•1    S
1•05000   B◊
     10    S
      2    S
2•10000   A◊
2•20500   A◊
2•31525   A◊
2•43101   A◊
2•55256   A◊
2•68018   A◊
2•81418   A◊
2•95488   A◊
3•10262   A◊
3•25775   A◊
```

```
       1•5    S
         1    S
       0•1    S
1•05000000   B◊
        10    S
         2    S
2•10000000   A◊
2•20500000   A◊
2•31525000   A◊
2•43101250   A◊
2•55256312   A◊
2•68019127   A◊
2•61420083   A◊
2•95491087   A◊
3•10265641   A◊
3•25778923   A◊
```

```
          1•5    S
            1    S
          0•2    S
1•10000000000   B◊
            5    S
            2    S
2•20000000000   A◊
2•42000000000   A◊
2•66200000000   A◊
2•92820000000   A◊
3•22102000000   A◊
```

CASE I

Figure 2



CASE I

Figure 3

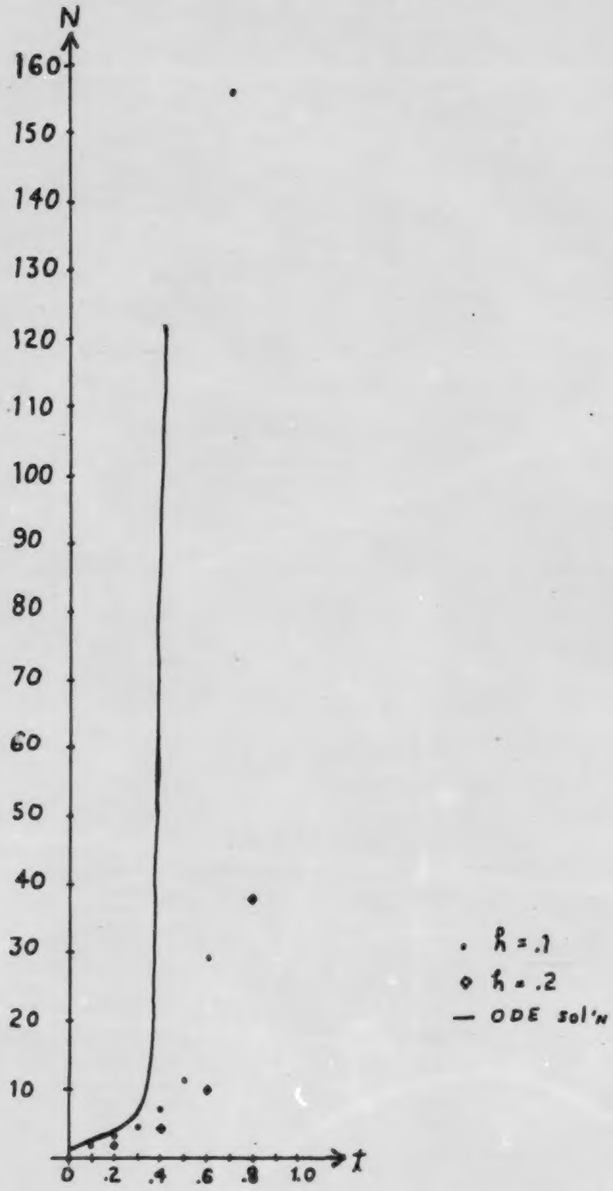
```
  V
  S
b ↑
 AY
e *
  S
B ↑
  S
c ↑
  S
C ↑
  ↓
 BX
B :
C ↓
c X
b -
c.↕
 B0
 c0
  S
d ↑
  S
D ↑
a W
D ↓
 BX
c -
D X
 A0
D :
e ↓
b +
e ↕
d ↓
e -
 /W
  Y
```

```
                  V
              1   S
            1·5   S
              1   S
            0·1   S
           0·15  B0
          -0·90  c0
             10   S
              2   S
           2·40  A0
           3·02  A0
           4·07  A0
           6·14  A0
          11·17  A0
          28·70  A0
         149·24  A0
        3474·30  A0
     1813723·57  A0
 49344060063l·76  A0

            1·5   S
              1   S
            0·1   S
        0·15000  B0
       -0·90000  c0
              9   S
              2   S
        2·40000  A0
        3·02400  A0
        4·09328  A0
        6·19718  A0
       11·33817  A0
       29·48740  A0
      156·96467  A0
     3836·95426  A0
  2211785·92330  A0
```

```
      1·5   S
        1   S
      0·2   S
     0·30  B0
    -0·80  c0
        5   S
        2   S
     2·80  A0
     4·59  A0
     9·96  A0
    37·64  A0
   455·06  A0

      1·5   S
        1   S
      0·2   S
  0·30000  B0
 -0·80000  c0
        5   S
        2   S
  2·80000  A0
  4·59200  A0
  9·99953  A0
 37·99671  A0
463·52224  A0
```

CASE II

Figure 4



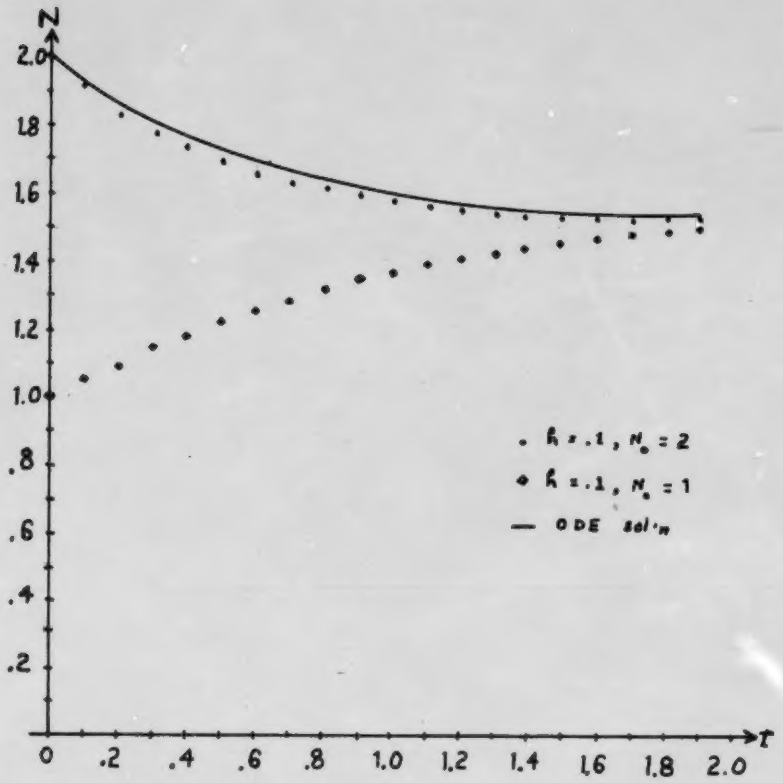
CASE II

Figure 5

```
AV            1  S           1·5  S             1·5  S
MS          1·5  S             1  S               1  S
b↑            1  S           0·1  S             0·1  S
AY          0·1  S    1·15000000  B0     1·15000000  B0
e*      1·15000  B0   0·10000000  c0     0·10000000  c0
MS      0·10000  c0           30  S              30  S
B↑           10  S             2  S               1  S
MS            2  S    1·90000000  A0     1·05000000  A0
c↑      1·90000  A0   1·82400000  A0     1·09725000  A0
MS      1·82400  A0   1·76490240  A0     1·14144174  A0
C↑      1·76490  A0   1·71814971  A0     1·18236908  A0
M↓      1·71814  A0   1·68066832  A0     1·21992478  A0
BX      1·68066  A0   1·65030397  A0     1·25405185  A0
b◆      1·65030  A0   1·62549925  A0     1·28493099  A0
B↕      1·62549  A0   1·60509936  A0     1·31256588  A0
c↓      1·60510  A0   1·58822987  A0     1·33716785  A0
CX      1·58823  A0   1·57421694  A0     1·35894124  A0
c↕      1·57422  A0   1·56253358  A0     1·37811030  A0
B0                    1·55276251  A0     1·39490804  A0
c0                    1·54456974  A0     1·40956740  A0
MS          1·5  S    1·53768563  A0     1·42231448  A0
d↑            1  S    1·53189076  A0     1·43336381  A0
MS          0·1  S    1·52700545  A0     1·44291520  A0
D↑      1·15000  B0   1·52288171  A0     1·45115205  A0
eW      0·10000  c0   1·51939709  A0     1·45824063  A0
D↓           20  S    1·51644991  A0     1·46433015  A0
cX            2  S    1·51395536  A0     1·46955340  A0
E↕      1·90000  A0   1·51184258  A0     1·47402769  A0
B↓      1·82400  A0   1·51005218  A0     1·47785609  A0
E-      1·76490  A0   1·50853426  A0     1·48112865  A0
DX      1·71814  A0   1·50724684  A0     1·48392374  A0
A0      1·68066  A0   1·50615456  A0     1·48630934  A0
D↕      1·65030  A0   1·50522759  A0     1·48834420  A0
e↓      1·62549  A0   1·50444073  A0     1·49007898  A0
b◆      1·60510  A0   1·50377265  A0     1·49155730  A0
e↕      1·58823  A0   1·50320533  A0     1·49281657  A0
d↓      1·57422  A0   1·50272350  A0     1·49388893  A0
e-      1·56253  A0
/W      1·55276  A0
MY      1·54457  A0
        1·53769  A0
        1·53190  A0
        1·52701  A0
        1·52288  A0
        1·51940  A0
        1·51645  A0
        1·51396  A0
```

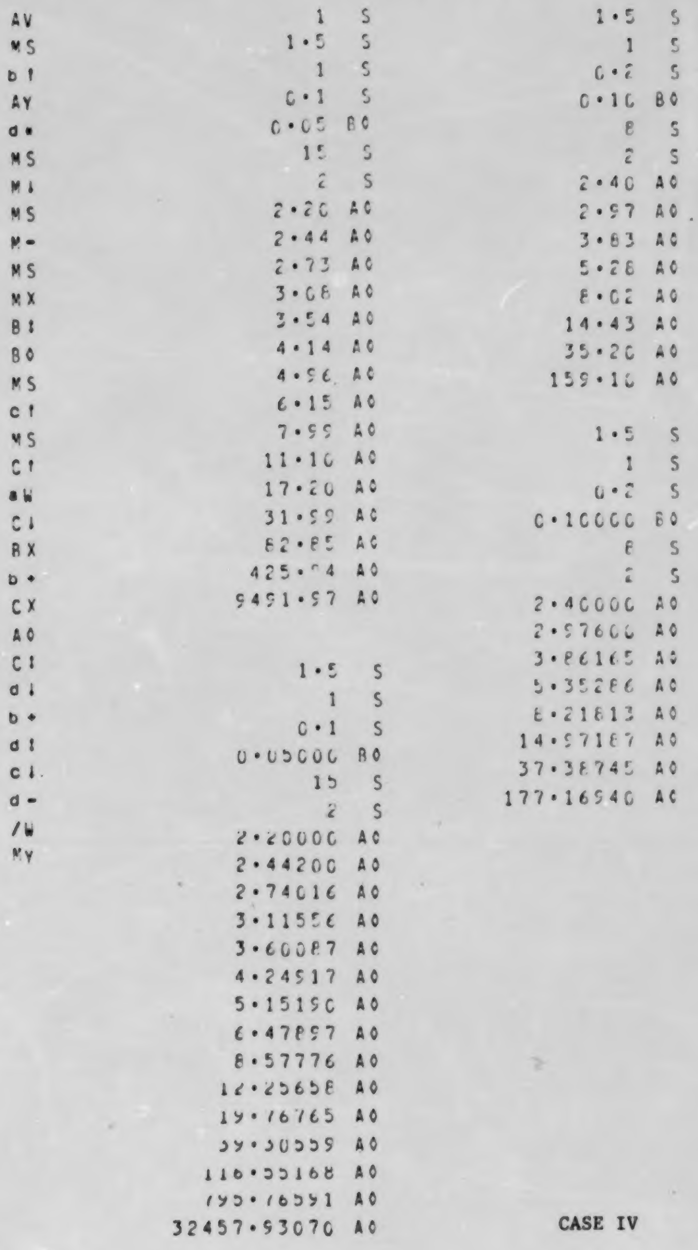
CASE III

Figure 6



CASE III

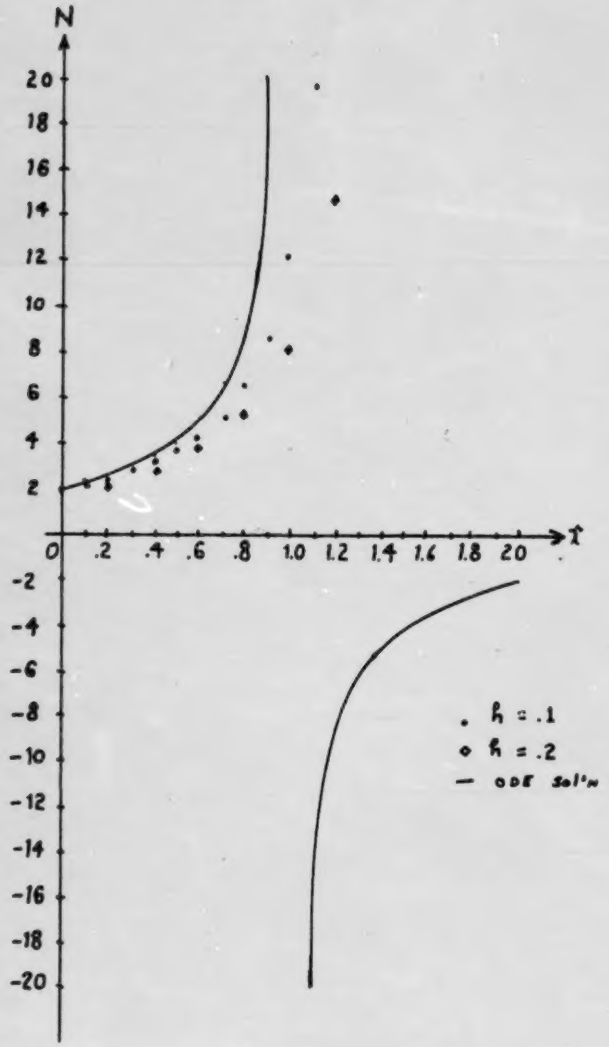
Figure 7



```
AV                  1   S          1·5   S
MS                1·5   S            1   S
b !                 1   S          0·2   S
AY                0·1   S         0·10  B0
d •              0·05  B0            8   S
MS                 15   S            2   S
M ↓                 2   S         2·40  A0
MS               2·20  A0         2·97  A0
M -              2·44  A0         3·83  A0
MS               2·73  A0         5·28  A0
MX               3·08  A0         8·02  A0
B ‡              3·54  A0        14·43  A0
B0               4·14  A0        35·20  A0
MS               4·96  A0       159·10  A0
c !              6·15  A0
MS               7·99  A0          1·5   S
C !             11·10  A0            1   S
■W              17·20  A0          0·2   S
C ↓             31·99  A0      0·10000  B0
RX              62·85  A0            8   S
b +            425·94  A0            2   S
CX            9491·97  A0      2·40000  A0
A0                             2·97600  A0
C ‡               1·5   S      3·86165  A0
d ↓                 1   S      5·35286  A0
b +               0·1   S      8·21813  A0
d ‡           0·05000  B0     14·97187  A0
c ↓.               15   S     37·38745  A0
d -                 2   S    177·16940  A0
/W            2·20000  A0
MY            2·44200  A0
              2·74016  A0
              3·11556  A0
              3·60087  A0
              4·24517  A0
              5·15190  A0
              6·47897  A0
              8·57776  A0
             12·25658  A0
             19·76765  A0
             39·30559  A0
            116·55168  A0
            795·76591  A0
          32457·93070  A0
```

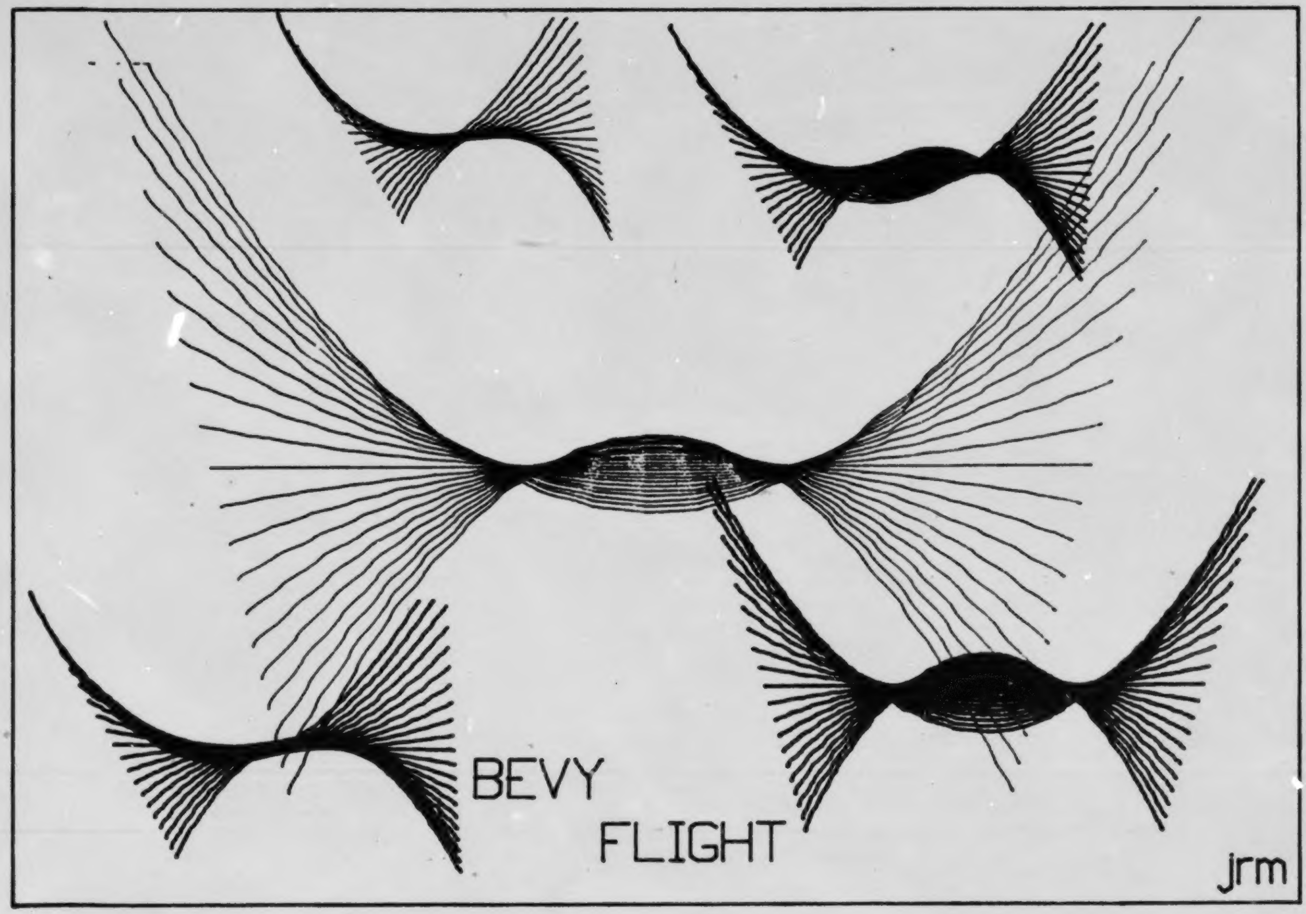
CASE IV

Figure 8



CASE IV

Figure 9


BEVY
FLIGHT
jrm

John R. Merrill

BUILDING A MACHINE-READABLE TEXT LIBRARY:
SOME OF THE VARIABLES

Ben R. Schneider, Jr.
Lawrence University
Appleton, Wisconsin 54911

A major obstacle to classroom use of computer methods in literary and linguistic study is the great expense and consequent scarcity of textual material in machine-readable form. Students must make do with whatever bits or scraps the computer center happens to have lying around. They cannot often follow their own interest in a particular topic, author, or language. The expense and scarcity of machine-readable text also dampens faculty enthusiasm for the application of computer methods in their own research. The result is make-shift courses in computer applications in the social sciences, humanities, and linguistics, in which dedicated and resourceful computer scientists do their best to suggest how their students might possibly use computers in the studies that interest them most. Lack of textual material in the area of interest, however, effectually prevents any actual application and development of the ideas suggested by the course. Lack of computer experience in the faculty at large, also caused by the same scarcity of machine-readable text, makes it harder still for the student to get the advice he needs from specialists in the field of interest he pursues.

The following paper does not suggest a solution to this problem. It attempts merely to lay down criteria of a good system for converting text to machine-readable form, and it measures seven methods of converting text against these criteria. A first step in assembling a library of machine-readable text is a solution of this mechanical problem. Such problems as finance, copyright, and distribution are not considered here. Greater progress may be made on these problems when some of the mechanical variables are more clearly defined. My conclusions are based on experiences gained in a pilot project with the task of finding the best way of converting the London Stage, an 8,000 page calendar of events at the theatres from 1660-1800, into a computer-accessible information bank for scholars in history, literature and drama.

Since it is impossible to predict in advance the kinds of analysis to which these machine-readable texts may be subjected, editions for a machine-readable library of literary and historical texts ought ideally to retain every bit of information that can be coded onto the magnetic tape. Not all information on a printed page is communicated by words. Type fonts, upper and lower case, punctuation, and page format, communicate various kinds of information. Format differentiates prose from poetry, for example. Capitals are clues to proper names. Italics usually signify titles. There is a qualitative difference between material in brackets and material in parentheses. In dictionaries, indexes, and lists, hierarchical relationships are shown by format and by type font. All typographical conventions exist because they convey information. Insofar as a system of data entry loses typographical data it loses information.

If the system involves keystroking (typing, keypunching), it must foster maximum speed with minimum error. Both goals are facilitated if instructions to the keystroker are as simple as possible. It would be best if she were asked merely to follow copy. But typewriters and keypunches cannot often follow the details of printed copy, because their character sets are too limited. A system of coding must be used to compensate for the deficiency. Some tagging of categories of material at this stage can save weeks of programming later and sometimes capture information that no amount of programming ever could capture. In a bibliography we might tag, for instance, author, title, publisher, place of publication, and date. The best system will enable a clear and simple common sense coding and tagging method that is easy to remember. There is no way to remember, for instance, that bracketed information begins with a plus sign and ends with an ampersand. "Greater than" and "less than" symbols opening towards the bracketed material would make much better sense, in the absence of real brackets.

The keyboard ought to have a character set that comes as close as possible to matching the character set of the text being converted, thus obviating the use of codes for missing characters or symbols. The device also should facilitate the entering of whatever codes are considered useful or necessary. It follows that the best keystroking device is the one with the most expressive keyboard.

A good input device enables the keystroker easily to correct accidental misprints that occur as she works. Ideally, she should be able to correct a mistake at once by striking over wrong letters, or inserting or deleting words or letters without having to rewrite any of the correct part of the text she has copied.

Any system needs a method for editing the magnetic tape after it is made. Even the exact magnetic tape "facsimile" produced by some scanners would probably need further editing. Texts which use format to indicate relationships of categories of information may fail occasionally to follow their own rules. How hard it is, for instance, to make all footnotes fit the straight jacket required by a style manual! Some rewriting of deviant material will be necessary before programs to parse such data can be written. It is usually easier to change a few places in a text than to write elaborate programs to deal with rare exceptions. Editing will be required also after scanning, if any tagging of types of items is needed to enable or assist future programming. If keystroking is used, much editing may take place before keystroking but there will be some mistakes to correct after the process is finished.

Since keystroking of literary texts is a time consuming process--it would take a typist about five weeks to do a six hundred page book--machines for converting such data should be dedicated to that purpose. The job cannot be done on time-sharing terminals that will not work unless the computer is in time-sharing mode, or on computer room keypunches that everyone else has to use.

There are at least ten ways of converting textual material to a machine-readable form. I shall not consider the key to computer tape device or the key to paper tape device or the online typewriter, because the first two suffer from disqualifying inflexibility and the online typewriter of course requires the attention of a computer.

The accompanying chart attempts to show how well the other seven methods of textual conversion measure up to the criteria I have laid down. I have graded them in each category of performance on a scale of five, five being a perfect score. These scores of course show only the relative position of each device to the others. The printer's magnetic tape method, for instance, could be so superior in the "error free" category that if it could be used, no other method might even be considered.

The standard *keypunch* has the most expressive keyboard of all the devices considered. Only by an elaborate, arbitrary, and unreasonable system of coding can it capture lower case, varying fonts and the punctuation that it lacks. Although the 029 keypunch is considerably better than the 026, an extra keystroke will be needed to denote every capital letter even with this machine. Tagging is difficult because so many symbols are used up in codes for typographical features that symbols having some mnemonic properties are hard to find. In process corrections are very difficult to make on a keypunch. Holes in cards cannot be erased. Every error requires a new card (still another set of codes *could* be introduced to nullify mistaken characters or words just after they have been struck but nothing can be done about earlier mistakes short of punching a new card). When keypunching is used, the more editing can take place before punching, the better. After the text is in the computer, any further editing would have to be accomplished by a program enabling one to delete, add, or change a character, word, or line with a batch of correction instructions. Editing by finding and repunching cards in a file is impossible when large numbers of cards are involved. Cards must be reduced to magnetic tape as fast as they are punched to reduce the chance of accidents, but this is an extra step and it, too, is error prone. Pre-editing will, of course, give the keypuncher one more thing to worry about. In the United States keypunching is also the most expensive way of producing literary text, which is not surprising, because it is the most difficult. There are keypunching services, however, in countries having low labor costs, like Hong Kong, which advertise verified keypunching at less than half the American price. The low price may more than compensate for any problems created in the output by the use of a keypunch and by operators unfamiliar with the language.

The *Magnetic Tape Selectric Typewriter* (MTST) has numerous advantages over the keypunch. This machine is simply an IBM Selectric Typewriter that records the same characters on its own magnetic tape that it types on a page. MTST tape may be converted to computer tape at nominal cost. The MTST character set is much less limited in its flexibility than that of the keypunch. Both upper and lower case are of course available. Any of the "bouncing balls" offered may be used. At a price prohibitive for any but the most mammoth operation, one may have a ball made to order. During the keystroking process one can correct mistakes simply by back-spacing and striking over. The new character erases and replaces the old. The tape can also play back to the keyboard, retyping automatically what has been previously keyed. This text as well as any insertions, deletions, and changes to it can be captured on another tape during the playback process.

Because this machine is especially designed to provide special codes for the IBM Composing Machine, *any* character on the keyboard can be used as a unique code symbol when a "prefix" key is depressed before the character is struck. The character then prints in red on the hard copy, giving clear and rapid proof that the code has been made properly. With such a system, it is easy to create codes having mnemonic features. But editing on an MTST is not as easy as a general description of the hardware would suggest. In order to transfer a tape plus revisions to another tape, one must properly actuate a bewildering set of control buttons, as well as the keyboard. Search and playback are a rather slow process, and it is impossible to know whether the corrections "worked" without playing back the new tape. Again, the best method of editing would be to standardize irregularities and enter tags before keying. After keying any further changes can be made with a computer program that follows instructions keyed on the MTST.

An MTST costs almost $10,000, but any typist can operate one with very little training as long as she does not have to edit. Labor costs might therefore be lower than for keypunching, especially if the text entry project is used to fill in lapses in the work load of an established typing pool. The MTST is not, however, the sort of machine a housewife can keep around the house for spare time typing.

Optical Character Recognition (OCR) has real advantages, either when direct scanning or a *retyping and scanning process is used*. Direct scanning, in the present state of the art is possible but it is not error free. Dissly Systems, of Easton, Pennsylvania, is the only company I know of who are devoting their energies entirely to the development of programming which will enable their scanners to read any type font. Other scanning services prefer to have material converted to their standard font before scanning. But Dissly programming is still in the development stage, and texts which use a large number of fonts (like dictionaries) require extensive programming. These scanners do provide for human intervention when the computer is stumped, but even so, most print contains so many irregularities that scanner output has to be thoroughly proofread and corrected. The result is that at present, unless direct scanning is cheaper than keystroking, it is not much better from the standpoint of error production. It does have some advantages in other respects, however. The magnetic tape produced by a scanner can be an "image" of the printed page. Every character, every font, and the location on the page of every character (in other words, the format of the page) is recorded. This means that it is a simple matter to automatically enter and tag categories of information indicated by font and/or format, such as, for example, various sections of a dictionary entry. Such tagging is part of the Dissly service. Normalizing irregularities and tagging material not indicated by font and format would have to be achieved by means of a computer editing program. No pre-editing could forestall any of this.

Retyping and scanning compares well with MTST conversion. The character set, it is true, is *limited to the one type* ball provided by the scanning service, and the only one I have seen contains too many accounting symbols, useless and meaningless in a literary context, and too few enclosing symbols like brackets, open and closed quotes, etc. It does not even have all the common punctuation marks. And only one font is available. Still, this type ball has much greater power of expression than the keypunch character set. Furthermore, mistakes made in the keying process are quite easily corrected by a system of rub-out codes which instruct the scanner to ignore the letter, word, or line preceding the code and to continue reading the text that follows the code. This method seems almost as good as the backspace and strike-over method of the MTST. Another feature is that one can easily convert any character to a neat, visible, and meaningful code by backspacking and putting a slash on top of it; the process is as simple and the result as good as those obtained by the MTST "red" coding system. Although no automatic coding is possible as it is with direct scanning, all tagging and normalizing can be done in the text before typing, and included in the scanning typescript. Changes made after scanning will have to be done by a computer program. But only those produced by accident will reach this stage. I have been quoted a price for typing and scanning that puts it in the same range as MTST conversion. Although the scanning costs must be added to the typing costs, a housewife *can* keep a Selectric typewriter around the house for typing in her spare time, whereas *the MTST* must operate with an office overhead.

The conversion of *printers' paper tape* solves and causes the same problems as direct optical scanning *except that under the right* circumstances the problem of error can be practically eliminated. The most modern linotype involves a two-step process. The composing machine, instead of producing hot metal lines of type at a rate that slows down the operator, produces a paper tape similar to teletype tape. It is a simple matter to convert linotype tape to computer tape, but the computer tape will be the image of the *galleys*, not of the published book. Instead of correcting punched paper tape, a very difficult thing to do, the printers correct the line of type, or, when the book is done by photo offset, paste in the corrections. If it is possible to obtain galleys and page proofs as a guide to where the corrections have been made in the published book, it may not be necessary to collate the computer tape with the book in order to complete conversion. As with scanning, the tape will be an image of the book and retain all the features of typography and format. But tagging and standardizing will still have to be done. Some of this tagging, of course, may be done automatically, by programs keying on type and format features, but probably not all of it. Since machines for converting linotype paper tape are inexpensive, the cost of conversion from galleys to computer tape would be nominal. Updating and coding the galleys would be the principal cost of this process.

Conversion of monotype tape, a by-product of a well established typesetting process more common in Europe than in the United States, would have precisely the same advantages and disadvantages as the conversion of linotype tape, except that there is one problem. The tape is incompatible with standard 6, 7, and 8 channel tape drives; it is over three inches wide. In this process a machine sets one piece of type at a time selected according to information punched on a tape by a compositor. Since the type-setting machine reads the tape as pulses of air coming through the holes in the tape to actuate pistons which in turn select characters, there are not even electrical impulses here to decode. However, it ought to be possible to devise a system in which the movement of the pistons in a monotype machine could be converted to electrical impulses to run a conventional paper tape punch. Converting monotype tape, then, probably would not be much more expensive than converting linotype tape. It might even be profitable if one happened to end up owning the only device in the world by which such tape could be converted.

If a book is printed by the new photocomposition methods, printers' computer tape, used to drive the photocomposition machine, is a by-product. This tape, of course, would be the best available digital image of the printed book in all its typographical details. It would even contain all the galley and page proof corrections, if, as is sometimes the case, they were made on the computer tape instead of being pasted in. Although printers' computer tape would probably require some post-editing, for the reasons already given, before becoming an edition in a machine-readable library of literary texts, it seems obvious that such tapes, if available, ought to be the final solution to the text conversion problem.

A Cathode Ray Tube Terminal (CRT) designed for data entry and text editing has properties analogous to the MTST. A CRT terminal, simply speaking, is a typewriter that types on a TV screen instead of on a piece of paper. What it types on the screen is readable by a computer. There are now half a hundred terminal manufacturers and seven of them make terminals that would work well in a system for establishing a machine-readable text library. These have upper and lower case keyboards with as much expressive power as an MTST would have if restricted to one type ball. The Imlac terminal, however, can itself be programmed. Its character set is even produced by software, and it can be altered by software. Imlac can produce the characters of any alphabet and use a repertoire of as many as 196 different characters. The facilities provided by these terminals for corrections during the keying process are truly marvelous. One simply moves the "cursor", a marker that indicates and determines where the next character keyed will appear, to the beginning of the place in the text that needs to be changed. At this point one may strike over the bad text, or after pressing a control button, delete or add characters, words, or even sentences. When material is deleted the empty space automatically fills with whatever text remains to the right of it, snaking up from lines beneath to do so. When material is added, a space is automatically opened up for it in the same way. No other device allows for insertion and deletion simultaneously on the page and in memory. This does so without harming the correct portion of the text. The MTST cannot insert and delete except in the process of transferring from one tape to another. Even then it is impossible to tell whether the correction process has worked. Although these terminals (excepting Imlac) provide no convenient way of turning any character into a special code, they have a "format" capability which will simplify coding in some applications (like bibliographies) where the same structure is repeated over and over again. The structure simply appears on the screen as a form to be filled in: "author---, title---, publisher---, place---, date---". The operator merely fills and empties this form over and over again. Each item is automatically tagged by virtue of its position on the screen when it goes into storage.

It would be a drawback if such a terminal could not be dedicated during all working hours to something as extensive as a text-conversion project. Terminals that cannot capture and edit text without the help of the main computer cannot therefore be considered unless the computer is always in time-sharing mode. Nor at the same time can those so-called "terminals" which cannot communicate at all with a computer be considered. A text entering and editing terminal should be able to transmit text when the main computer storage is ready to receive it and to receive text for updating from the main computer storage. By using their own tape cassettes for input and output and performing all their text editing functions by means of their own hardware or software these terminals can enter or edit data whether or not the main computer is available to help them. They also communicate with the central processor like teletypes. Thus they maintain their independence without giving up their connection with the big computer. Editing on such terminals is almost as easy to do as editing texts on a piece of paper. It is probably harder to position a cursor, push a function key, and key in changes than it is to make a few marks on a page with a pencil. But the pencil marks must be interpreted and copied by someone before they enter a computer. Post-editing on a CRT, therefore, instead of pre-editing with a pencil, eliminates one error-producing step, and it may not be much harder. At the same time it reduces the number of details the typist has to worry about. Certainly post-editing with a CRT is the best way to complete tagging and standardizing when conversion is by the direct-scanning or printers' tape methods. Indexing by automatically reading out to the computer the position of a cursor or light pen, placed on the word in the text desired for the index, is also a possibility with two of these terminals.

Obviously, such a device would also be an ideal tool for literary and linguistic analysis using computers. Suppose, for example, one wished to study a writer's syntactical habits. The ideal way of labelling the parts of speech in samples of his work would be to label the words as the text passed across a CRT screen, identifying them and entering the identifying information into the computer in a single step.

The Imlac display terminal costs a bit more than an MTST. The other terminals that are discussed here cost about two-thirds as much. The Imlac, however, does a great deal more than the others. Besides being programmable, it does the curves, figures, and graphs desired by the mathematicians and scientists. With this machine, a well understood system of priorities among users would be mandatory.

FOOTNOTES

1. For an interesting account of the keypunch's slow release of its uncanny hold on the data-processing industry, see Robert C. Stender, "The Future Role of Keyboards in Data Entry," *Datamation*, November, 1970.

2. Well-covered in an article by R.S. Morgan, *Computers and the Humanities*, November, 1970.

	KP	MTST	OCR	T-OCR	PPT	PMT	CRT	IMLAC
Character set	1	3	5	2	5	5	2	5
In Process Corrections	2	4	-	3	-	-	5	5
Ease of Coding	1	4	4	4	2	2	3	5
Edit, update	2	3	2	2	2	2	5	5
Error production	2	3	4	3	3	5	3	3
Average	1.6	3.3	3.7	2.8	3.0	3.5	3.6	4.6

KP - Standard Keypunch
MTST - Magnetic Tape Selectric Typewriter
OCR - Optical Character Recognition
T-OCR - Typing and Optical Scanning
PPT - Printing Paper Tape
PMT - Printing Magnetic Tape
CRT - Cathode Ray Tube Terminal
IMLAC - The IMLAC CRT

COMPUTER-AIDED ANALYSIS OF MUSIC USED IN UNDERGRADUATE MUSIC THEORY

R. N. Killam
Tarleton State College
Stephenville, Texas

INTRODUCTION

The concepts taught in the first two years of music theory, e.g. harmonic, melodic, and rhythmic relationships, are, for the most part, simple and rather precisely defined. However, music theory is considered a difficult course. One of the main sources of difficulty is the large number of concepts which must be learned thoroughly and related one to the other.

In order to relate concepts under study to the whole of music, the instructor must have at hand a representative number of examples of each concept, taken from standard musical literature. Textbook authors generally illustrate each concept with fragments of a few measures only from standard literature, and give references at the end of each chapter citing examples for further analysis. Often, fragmentary illustrations lose their musical validity out of context, and most instructors prefer to have at least a few musical works available in their entirety to each student. Most of these works have been analyzed for elementary concepts by the instructors in the course of their own studies, but little cross-indexing by concept is readily available.

Students must learn to identify each concept in the analysis of musical literature, and must learn also to apply concepts to performance and composition. The hierarchical ranking of concepts in their relationship to the whole of a musical structure is clarified by some simple calculations, the performance of which is greatly accelerated by computer.

Although much sophisticated computer programming is utilized presently in synthesis and analysis of music (1), simple programs can be of value in introducing students to the computer. Also, even relatively simple programs can save much time for teachers in lesson planning. Since musical analysis is largely a problem of data sorting, storage, and retrieval, use of such data processing facilities as are generally available on most college campuses can greatly facilitate analytical procedures.

This paper describes an attempt to use the computer in undergraduate music theory courses. Programs have been developed which students and teachers can use in the analysis of music. In addition, the programs were found to be of value in introducing students to the computer.

MATERIAL COVERED

The area selected for an introductory teaching unit utilizing computer-aided analysis was that of melodic construction. Students were taught analysis of scale tones and the intervals between scale tones. Each student was assigned a simple melody to analyze from the Ottman Sight Singing Manual (2). One class session was devoted to explanation of keypunch machines, and sample data cards were punched. Then, each student keypunched this data, one measure to a data card. These cards were submitted for proofreading, and errors in punching corrected. Then, each student was given a deck containing a program previously written to calculate percentages of scale tones and intervals. The programs were run by the data processing center.

Student enthusiasm was much higher than expected. All material was turned in on time and 15 of 19 programs ran without any errors on the first attempt.

Students immediately saw further applications, not anticipated by the instructor, such as using the same type of analysis to implement reports being made in a separate class in music literature. Also, several students plan to employ random generation of pitch, interval and rhythmic sequences in semester composition projects.

The experience generated considerable interest in taking computer science courses to fulfill degree requirements in the math-language area. The typical reaction seemed to be, "I want to be able to write my own programs to do this kind of stuff".

The harmonic concepts taught in the first two years of theory are usually presented in grouped sequence. One of the standard assignments given to students to reinforce understanding of a concept is analysis of a given number of musical illustrations containing the concept.

The 371 chorale harmonizations of J. S. Bach are one of the standard music literature resources for the teaching of elementary theory. Bach maintained a clearly-defined compositional style throughout the chorale harmonizations, and used a wide range of harmonies. The chorales, therefore, are an invaluable aid in teaching basic tonal harmony, and are used to illustrate almost all systems of teaching elementary music theory. However, Bach's use of varying diatonic and altered harmony, modulations, non-harmonic tones, etc., endows each chorale with differing harmonic characteristics. Most theory texts give listings for analysis of only five to ten chorales containing the harmonic concept under study, although the particular harmonic device may be present in over a hundred different chorales. Also, the concept may be present in conjunction with material not yet covered by the student, so that the example may be too complex for student analysis.

Rare is the teacher who has memorized all the 371 Bach Chorales. A beginning theory teacher may not have worked with the chorales in some time, and is faced with the daily necessity of designating chorale excerpts of sufficient simplicity for student analysis. Obviously, if the analytical data of the chorales were to be gathered together, sorted under headings for single concepts and clusters of concepts, a great deal of time spent in class preparation could be saved.

METHODS

Data for single concepts can be accumulated in a single hand-drawn matrix, but concept clusters, especially in conjunction with the absence of more advanced material to be presented later in the course, is too complex and time-consuming to sort by hand.

Therefore, an appeal for help was made to the computer center. Early in consideration, direct encoding of music was abandoned as a technique. The computer languages which are developed and suitable for musical analysis are not generally available, and the programming personnel with whom the writer has had contact are not familiar with them (3).

The writer is in the same dilemma as many musicians--conservatory training generally does not include instruction in data processing. For the present, study and use of the computer seemed best limited to langauges with which help could be obtained.

Two other considerations militated against direct computer analysis of the music, at least for the present:

1. present analytical programs are costly in computer time;
2. any body of musical literature envisaged as a standard resource for elementary music theory has already been subjected to extensive, if not total analysis. Analytical results may be coded easily for submission as part of computer data.

On the other hand, even if the material is completely pre-analyzed, relationships which are difficult to check and easily overlooked become obvious once the data is computerized. In addition, the data can be made available easily to other researchers in related studies. Therefore, the decision was made to concentrate on coding hand-analyzed data.

The edition of the chorales selected was that of Mainous and Ottman (4). The edition is readily available and is priced within student reach. It contains all chorales in their traditional order, and is already cross-indexed in several ways. Dr. Ottman and Mr. Mainous were consulted prior to coding; both gave permission to use their edition. Dr. Ottman, in addition, suggested several different categories of data to be tabulated.

Four computer programs were written to perform various analyses. All programs are written in FORTRAN IV and none is over one hundred statements in length. The programs accomplish the following functions:

Program 1 calculates percentages of scale tones and intervals in one-line melodies. Percentages of interval usage are printed in matrix form so that any incorrect analysis may be spotted easily.

Program 2 checks data deck for chorales to insure proper card sequence, since total data deck will run in excess of 1,000 cards.

Program 3 tabulates chorale characteristics by comparison of simple absence or presence and builds a table listing each characteristic and the numbers of the chorales containing it. This process continues sequentially through all characteristics. The program is written to allow additional characteristics to be added to data.

Program 4 has varied form for checking for absence or presence of any group of the coded characteristics. It sums the number of chorales and compiles and builds a table of the chorales of the given configuration.

Copies of programs will be made available at cost.

CONCLUSIONS

Many people, both in and out of the arts, regard with suspicion any mechanized encroachment on music. But the fact remains that a great many of the basic concepts in music are clearly defined and repetitively used.

Musical study has always proceeded by orderly transmission of this heritage from teacher to student--thus Beethoven studied under Haydn; many good musicians can trace a direct lineage of study to celebrated classical composers. Basic historical concepts must be mastered by students so that they can proceed knowledgeably in individual endeavors in more contemporary areas. The acquisition of basic background has always been a matter of concentrated, tedious study. Analysis by computer of any of the mainstream musical styles will benefit elementary music theory.

The Bach chorale harmonizations fall within this category. Even though direct coding of the musical notation into the computer for analysis is still experimental, hand analysis of music and submission of the data to the computer makes possible sorting, tabulation, and search for new relationships. Although coding of the Bach Chorales is still in progress, data available on partial number of chorales has proved helpful.

The writer considers computer-aided analysis of other bodies of literature, including the J.S. Bach Two-Part Inventions, Well-Tempered Clavier, Beethoven Piano Sonatas, Chopin Preludes and Mazurkas, to be valuable undertakings. All of these are drawn upon constantly by teachers and students for examples of basic compositional techniques and concepts. Course work which utilizes such literature would be greatly facilitated by computer-aided analysis; most teachers cannot quote all occurrences of any given concept in a given body of literature, no matter how well they know it.

In addition, introduction to computer techniques can be accomplished very early in undergraduate studies. Analysis of such simple concepts as melodic usage of scale tones and intervals can be introduced in the first term of first year music theory. Such computer-aided analysis takes relatively little class time and amplifies the students' understanding of the concepts. Students take to such projects with great enthusiasm, a commodity sometimes in short supply in basic study.

ACKNOWLEDGMENTS

The writer would like to acknowledge the interest and assistance of Clair B. Smith and Robert Nehs, both of whom have provided advice and programming assistance. Thanks are due also to Linda McElroy for aid in research and typing. The programs were written while the author was participating in a faculty development program conducted by the Texas Regional Academic Computing Experiment. Dr. Roger Elliott, director of TRACE, has given generous amounts of time to direction and encouragement of the author's endeavors.

REFERENCES

1. Erickson, Raymond, "A General-purpose System for Computer-Aided Musical Studies", *Journal of Music Theory*, Winter, 1969, Vol. 13, No. 2, pp.276-294.

2. Ottman, Robert W., *Music for Sight Singing*. Englewood Cliffs, N.J., 1967

3. Winograd, Terry, "Linguistics and the Computer Analysis of Tonal Harmony", *Journal of Music Theory*, Spring, 1968, Vol. 12, No. 1, pp. 2-49.

4. Mainous, Frank D., and Ottman, Robert W., *The 371 Chorales of Johann Sebastian Bach*. New York: Holt, Rinehart and Winston, 1966.

TRAINING THE HUMANIST FOR HIS ROLE IN DEVELOPING UNDERGRADUATE COMPUTER ORIENTED COURSES

Floyd R. Horowitz
University of Kansas
Lawrence, Kansas 66044

The presentation is concerned with a perspective of the commitments necessary to the training of humanities faculty in the skills and technology of computers. Training programs supported by federal and private agencies are discussed, as are network arrangements between colleges designed to develop background facilities for undergraduate humanities computing. Emphasis is put upon the typical initial problems faced by the humanist who desires to gain a competence in computing that would allow incorporation of such techniques together with traditional curricular materials. Specifically, the program of the National Summer Training Institute for Humanities Computation is reviewed as a possible model for adoption by universities and colleges.

(The author will participate in a panel discussion at the conference in which he will expand upon these ideas.)

A COMPUTER-BASED INTRODUCTORY PHYSICS COURSE

Ronald Blum
Commission on College Physics
University of Maryland
College Park, Maryland

Introduction

The purpose of this paper is two-fold: (1) to state--and demonstrate--the pedagogical reasons for computer usage, and (2) to describe a proposed computer-based introductory physics course. The viewpoint taken here is not based on learning theory, but is entirely discipline-oriented, the approach to the computer being through its application to problem solving and to the simulation of physical reality. The course itself would attempt to incorporate the computer into the classical two-year curriculum for physics and engineering majors by embedding a course-within-a course into the usual structure of introductory physics. This subsidiary course would cover the fundamentals of numerical analysis, algorithm construction and numerical methods in the context of the physics and so underscore what the author conceives to be its essential unity with physics.

While the course is predicated upon the availability of only minimal computer facilities, such as a time-shared teletype and the BASIC language, there is no attempt to discuss computers or computer languages per se within the course. Instead, flowcharts are freely employed as a universal language, and the mathematical details are expressed in the ordinary language of algebra. In principle, one does not even need a computer to study the methods with profit.

Although various portions of the course-within-a course have been well-known and class-tested, the course as such has not been taught to students. This paper constitutes a proposal and an argument that such a course is needed and viable.

Educational Computer Usage

We can distinguish at least five modes of educational computer usage, each embracing several types of output (alphameric or graphic, paper or film, temporary or permanent). The five modes are (1) producer, (2) administrator, (3) tutor, (4) simulator and (5) calculator, listed roughly in order of increasing demands on the students' understanding and participation. Although these distinctions are not mutually exclusive, they are useful.

In the first mode, the computer can be used to produce materials such as illustrations, films and textual materials (through the use of formatting and editing programs). As an administrator the computer can give exams, grade them, recommend individually-tailored reading assignments to students based on their grades, give students information about their performance in the course, and carry out onerous clerical chores connected with teaching. A computer-managed instruction (CMI) system can assist the instructor by providing him with diagnostic information for making decisions concerning course content, structure, lesson schedules, supplementary materials, grouping of students on various "tracks" and prescriptions for individual assistance.

In the tutorial mode, the computer is programmed to converse with the student, give him information, ask him questions, and react to his responses. This mode, which resembles programmed instruction and lends itself to reviews, drills, and self-instruction, has been much admired by educational researchers, especially because of its potential for returning vast quantities of data for evaluation of performance. However, it is still very costly, variously estimated at $2.00 to $80.00 per student hour. Also, anywhere from 10 to 200 hours of programmer's time may be necessary to produce tutorial material sufficient for one hour of terminal time, the lesser estimates being for drill and review programs.

It is the simulator and calculator modes of usage which are most applicable to physics and science generally, on all levels. When used in the simulation mode the computer becomes a device to simulate physical reality in the form of governing equations, laboratory experiments or stochastic processes. This mode is inherently experimental and, if made the major component of the introductory physics laboratory can serve as an invaluable complement to the calculator mode which, in my view, should be the primary use of the computer in physics education. Hence, the emphasis in this paper is on the last mode.

In attempting to organize and classify the arguments for computer usage, the following (by no means exhaustive) categories come to mind:

I. Practical Considerations of Computer Usage

- A. Ease of computation: enables the student to do more physics with less mathematics.
- B. Allows treatment of more realistic situations and necessitates the careful study and appreciation of errors.
- C. Introduces algorithm construction, recursive procedures, numerical analysis, scaling and similarity arguments into introductory physics.
- D. Allows the simulation of experiments otherwise too costly or difficult to perform, and the "experimental" demonstration of concepts otherwise difficult to visualize.

II. Conceptual Considerations of Computer Usage

- A. Introduces precision and discipline into problem solving.
- B. Leads to greater awareness of our own logical processes of abstraction and analysis.
- C. Raises fundamental questions about the validity of the basis of our description of physical reality.
- D. Introduces a sense of independent, original discovery into even the most well-trodden problems.
- E. By complementing the traditional analytic methods, the computer broadens the physicist's problem solving repertory and lessens his psychological dependence on traditional methods.
- F. Greater awareness of computer-based methods and what they can achieve will make the student less psychologically subservient to the computer.
- G. Makes science and scientific concepts more readily accessible to non-science majors.
- H. Introduces "open-endedness" into the physics curriculum, homework, and laboratory.

We are often kept from many interesting problems in introductory physics by the students' lack of mathematical sophistication. With numerical methods, Newton's Second Law becomes a matter of careful bookkeeping, and even an elementary, non-science major can learn to do problems in harmonic oscillation, damped oscillations,[1] resistive flow,[2] etc. It is no more than basic arithmetic, either, to teach him to do a table look-up with linear interpolation by computer, so that he can use realistic data in his computations. The results are often suprising (see below).

Furthermore, there is much of a fundamental nature involved here, and through computer usage one learns things that cannot or will not be taught any other way. In physics, for example, computer usage naturally introduces more precise notions of round-off, truncation, and experimental errors; whether certain procedures are unstable, relatively stable or absolutely stable in regard to error accumulation; which processes introduce cumulative errors and which are self-correcting. It also leads to algorithm construction, recursive procedures and the study of numerical analysis, as well as an early and natural appreciation of scaling and similarity arguments. The student also receives the benefit of the discipline and precision required to communicate with the machine effectively and efficiently.

In the simulation mode the computer may, for example, simulate a mass spectrometer[3] or a bubble chamber.[4] The student does not need to understand the nature of the computer program. He merely inputs information and receives output which the computer derives by mathematical simulation of the physical situation. The real fascination of the simulation mode is its experimental nature. A sort of super-Gedanken experiment come to life. A student can design and perform his own "experiments" which may be as simple as a trial-and-error approach to solving some problem by varying inputs and studying outputs, or as refined as modelling a process like radioactivity, thermal diffusion, random walk, etc., through the use of random number generators which are to be found in every function library or are easily generated (providing a nice tie-in with the algebra of numbers modulo x).

It was interesting to note that in the Conference on Computers in Undergraduate Science Education (COMUSE)* the majority of demonstrations occurred in that third of the conference devoted to simulation. Furthermore, interactive computer graphics, analog computing and computer-animated films all lend themselves particularly well to simulation. Graphical representations, especially those providing immediate man-machine interaction are the key to the success of this mode. This is direct perception as in real life, without the intermediate step of digitizing your intuitions to numbers and symbols.

Another advantage of simulation is that one can "simulate" quantities which exist only in a mathematical sense such as electric or magnetic field lines, streamlines and "streak" lines. To carry this still further, while the simulation of a mechanics laboratory might allow the student to work in a universe where Newton's laws of motion hold, it might also allow him to specify non-Newtonian laws. Or the simulation can present the student with possibilities of experience which are not available normally, even though the laws are those of the "real" world; e.g., a simulation of the space-time viewpoint of relativity in which the student is given a 2-D space-time world on the face of a CRT. He can interact dynamically with the space-time diagram, inputting events and worldlines, and calling on Lorentz transforms.[5]

In a conceptual sense there is some tendency to consider the computer as "impure," because it necessarily deals with discrete quantities and approximations. If you are searching for pure Platonic essences, then you may look down your distinguished proboscis at the computer chugging along in its approximate way. But then you should also be prepared to discard decimal representation of all reciprocals of primes, except 2 and 5, and only use pure fractions. If your emphasis is on Man and his destiny, then the goal is description by any means. There is no inherent superiority in looking your sines and cosines up in a table as opposed to computing them on-the-spot. The methods used are the same: series approximations, for the most part.

The computer may also serve to free us from the ritual pace of the traditional curriculum. For example, everyone studies the principle of the direct-current motor, in which a current loop revolves in a constant magnetic field. We do not venture to analyze the rotation of such a loop, because the differential equations are too hard to solve, since the torque on the loop, hence the power drawn from the battery, hence the current in the loop, hence the torque, etc., depend upon the angular displacement. A detailed analysis, using a computer, is not beyond first-year students and would reveal that the rotational inertia of the loop can, at times, give rise to a back e.m.f. greater than that provided by the battery of the motor. This, in turn, gives rise to the question: what happens if an external torque forces the loop to rotate still faster? And so, in a sense, we can rediscover Faraday's Law for ourselves in a way which, though more complicated than the usual example of a loop with a sliding wire, is more natural an outgrowth of the student's earlier work. And much more provocative, since it catches him unawares.

It has been suggested[6] that since computers can now perform symbolic manipulations of algebraic quantities such as $(x-a)(x+a) = x^2-a^2$, many of the routine mathematical skills which we teach can one day be "mechanized" and omitted from an already lengthy and demanding curriculum. This poses the question whether or not we are perhaps teaching our students the wrong skills. However, I personally feel that the computer is destined to supplement, not supplant, analytical techniques, that the two must be used in a complementary way. We want to stimulate the student to learn more, not make it possible for him to do less.

An example of this is the horizontal motion of a spherical body of radius R through a resistive fluid.[2] The equation of motion is

$$m\ddot{x} = -\tfrac{1}{2}C_D \rho V A \dot{x}$$

known as Newton's resistance law, where V = speed, A = cross-sectional area, ρ = density, C_D = drag coefficient. The crucial parameter is the Reynolds number $N = \rho V(2R)/\eta$, and

$$C_D \doteq 0.5 \text{ for } 1{,}000 < N$$

and we have Newtonian flow, drag proportional to V^2. For $N < 10$ we begin to go over to Stokes flow, where $C_D \dashrightarrow 24/N$ and the drag becomes proportional to V. The peculiarly interesting result is that integrating analytically for pure Stokes flow and pure Newtonian flow yields a stopping distance of $x_\infty = 833M/\rho A$ (Stokes), $x_\infty = \infty$ (Newton); but in the realistic case

$x_\infty = 19.5M/\rho A$. Upon further reflection, one realizes that at low velocities the Newtonian drag becomes too small to stop the object, while at large velocities, the Stokes drag is relatively small. The realistic method utilizes both approximations where they do the most good. This also demonstrates just how misleading a complete, but approximate, solution can be, and how both pencil and computer play a part in this analysis.

Another aspect of computer use which this problem illustrates is its "open-endedness." After solving the horizontal problem, the student can go on to consider vertical motion in the atmosphere, where density varies with height, and finally the re-entry problem for a space vehicle, illustrating two-dimensional motion and the vector nature of the resistive force. This is one which it is difficult even to approach analytically.

Finally, we can illustrate Blum's Paradox: the more you understand about computers, the less dependent you are on them. As a trivial example of this, consider the problem of cooling 5kg of lead, specific heat (0.03+0.00002 T) kcal/kg.°C, at 300°C by immersion in 1 kg of water at 25°C. Ordinarily, we would ignore the temperature dependence of the specific heat to obtain a final result, T = 70°C; an error of 5.5°C from the correct answer T = 64.5°C. However, the alternative is an ugly quadratic:

$$f(t) = 0.00005\ T^2 + 1.15\ T - 74.5 = 0$$

with such a small leading term that straightforward slide-rule evaluation of the conventional formula

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

is hopeless.

The obvious solution is to run to the calculator or teletype; or, if he is clever, the student may expand the discriminant as a binomial and retain the first term, yielding the linear solution $x = -c/b$, $T_o = 64.8°C$ in this case. However, if he has learned to think recursively, he would immediately take T_o as the first guess to Newton's approximation

$$x_{i+1} = x_i - f(x_i)/f'(x_i)$$

which quickly and painlessly gives

$$T_1 = T_o - \frac{0.00005\ T_o^2}{0.0001\ T_o + 1.15} = 64.6^o\ .$$

And if he is aware of the second-order convergence properties of Newton's approximation, comparison of $T_1 - T_o$ with the results of a second iteration $T_2 - T_1$ will tell him approximately how far he is from the correct answer (64.5°C). The moral is: a knowledge of computer-oriented methods enables one to cope with more realistic problems, while at the same time reducing one's psychological dependence upon the computer itself, and also helping to avoid the trap of substituting number-crunching for insight.

The Course

I think that the best way to consistently and coherently introduce the computer into the curriculum is to have a text which does so for an entire course. The incorporation of a course in numerical methods and algorithms into the traditional curriculum is essentially without precedent, and early texts will surely need to be revised in the light of subsequent teaching experience. In the text which I have written computer-oriented methods are completely assimilated to the subject matter and there is nothing specifically on computers *per se*, or even on programming languages (although there will be an appendix on BASIC). Everything is done by means of flowchart "hieroglyphics" in standard algebraic notation. The methods taught always arise in the context of particular physics applications, thereafter becoming available for use in other parts of the course. The emphasis is not on computer usage, but on recursive methods; where these are amenable to hand computation this is forcefully indicated to the student. The goal of the course is a better understanding of, and capacity for, the description of physical reality; *not* computational gymnastics.

The course is on the freshman-sophomore level, with a requirement of concurrent calculus study, and a prerequisite of high-school algebra, geometry, and trigonometry. Previous or concurrent exposure to chemistry is helpful, but no prior computing experience is necessary.

My approach to integrating the computer was first to list those techniques which I felt desirable for a student of physics or engineering to know by the end of his first two years. These were organized as a short course and then embedded into the traditional curriculum as a course-within-a-course as follows:

TOPIC	CONTEXT
Numerical Differentation (first order)	Kinematics: velocity and acceleration. Deriving forces from position and time data.
Numerical Integration by Euler's Method (first order), Scaling, Similarity	Harmonic oscillator. Introduces recursive procedures. $m\ddot{x} = -kx$ becomes very naturally $(mL/T^2)u''=kLu$ and can derive independence of amplitude, estimate of time-scale; the motive being supplied by simplification of computation.
Trapezoidal Quadrature (first order)	Work, $W = \int \underline{F}.\underline{dr}$.
Two-Dimensional quadrature (Riemann Sum) Subscripted Variables, wArrays, Table Look-up.	Centroid of a plane surface.
Numerical Integration by the Improved Euler Method (2nd order)	Computation of elliptic orbit in inverse-square law field. Accuracy very high; even with $\Delta\theta = 45$ no worse than 12% error.
Newton-Raphson Method (2nd order), for Root-finding	Ascent of high-altitude balloon ("Skyhook") to the point where the adiabatically-cooled helium no longer provides any lift in the thin atmosphere (hydrostatics). Result from realistic assumptions gives altitude 30km (actual, 33km).
Simpson's Rule (3rd order)	Period of a real, physical pendulum. An integration simple enough to be performed by hand can give six-figure accuracy, as well as a refreshing feeling of mastery.
Least Squares Approximation, Averages, Simple Linear Regression	Interference. Fourier Series approximation shows that simple waves can combine to produce waves of any shape whatever. Leads into notions of curve fitting and "goodness of fit." Useful preparation for kinetic theory to come later.
Lagrange Form of Unique Interpolating Polynomial	Introduced with virial expansion of equation of state and Taylor's Series. Amenable to hand computation. Connection with Trapezoidal, Simpson's Rules. Derive, in example, interpolating polynomial for period of a physical pendulum over $0^\circ < \theta < 150^\circ$ to third order; accuracy better than 1.7%.
Random Errors, Monte Carlo Methods, Simulation of Random Processes	Kinetic theory. Introductory section on data reduction leads to heuristic derivation of Maxwellian distribution in terms of gaussian.
Integration of Elliptic PDE by Relaxation	Electrostatic potential computation (two dimensions).

Higher-Order Numerical Differentiation, using the Lagrange Interpolation Polynomial	Finding the capacitance of the condenser used in the relaxation calculation from the surface charge density $\epsilon \, \partial V/\partial n$.
Simultaneous Linear Equations, Determinants	DC Circuit theory
Simulated Experiment	Magnetic media. Finding the flux density in the gap in a permeable Rowland ring given magnetization data $\mu(H)$ and the magnetizing current. Can solve for $I(H)$ so we "experiment" with different H, using the computer as our ammeter and find $H(I)$ from the data.
Predictor-Corrector Methods of Integration, Error-Control and Stability	Faraday's Law. Motion of a damped ballistic galvanometer. Method developed by refinement of improved Euler method.
Complex Analysis	AC Theory. Computational definition of complex variable operations is stressed, so that we can incorporate the methods of DC theory, replacing $d/dt \dashrightarrow j\omega$, $\int dt \dashrightarrow 1/j\omega$.
Integration of Hyperbolic Wave Equations	Used in treatment of waves on transmission lines in connection with a discussion of initial and boundary conditions.
Numerical Integration of the Eigenvalue Equation	Schrodinger's Equation, square-well potential.
Newton-Raphson Solution of Simultaneous Transcendental Equations	
Newton's Forward-difference Interpolation, Stirling's Series (analogous to Taylor's Series)	

As can be seen, some of the examples flow quite naturally, some are rather strained, and some are frankly a gamble on the student's ability to undertake some hard thoughts with the aid of the computer. Therein lies the fascination--it is not all that easy, and there is a continual process of unfolding interrelations which is most intriguing. Given the computer, certain applications are fairly obvious. What is less obvious, however, is how the computer may ultimately alter the style and content of science education. The computer has an immense potential for changing the way in which we view the universe and describe it, and the way in which we organize our own processes of abstraction and analysis. In fact, the computer is not just a super-slide-rule but a revolutionary new force in our economic, social, and intellectual life; that challenge must be met by us, now. Even our liberal arts majors must have enough experience to know that "the computer says so" is not sufficient reason for accepting the decisions of government and industry. And also so that they are not too diffident to apply the computer to their needs.

Ultimately, computer usage will be fully assimilated within each discipline, and all such invidious distinctions as "computer-based" and "computer-oriented" will wither away. The student will be free to explore a broader range of more meaningful phenomena than ever before, and if we match his enthusiasm with our own vigorous dedication to the expansion of the human intellect by any means possible, I expect to see some exciting developments in the future use of computers in college education.

REFERENCES

1. Bork, A. M., A. Luehrmann, and J. W. Robson, Introductory Computer-Based Mechanics, R. Blum, Editor, Commission on College Physics 1968.

2. Blum, R., "Opening the Other End," in Computer-Based Physics, an Anthology, R. Blum, Editor, Commission on College Physics, 1969.

3. Harbron, T. R., and C. W. Miller, "Computer Simulation of a Mass Spectrometer," ibid.

4. Mikkelson, R. C., "A Simulated Accelerator Laboratory," ibid.

5. Taylor, E. F., "History of a Failure in Computer Interactive Instruction," in Proceedings of the Conference on Computers in Undergraduate Science Education, R. Blum, Editor, Commission on College Physics, 1971.

6. Engelman, C., "The Mechanization of Mathematics," ibid.

* Held at the Illinois Institute of Technology (IIT) in Chicago, August 17-21, 1970, and sponsored jointly by IIT and the Commission on College Physics.

USING A COMPUTER INSTEAD OF CALCULUS IN AN INTRODUCTORY PHYSICS COURSE

Donald L. Shirer
Valparaiso University
(Now at Computer Education Research Lab, University of Illinois)

Three years ago at Valparaiso University, it was decided to eliminate calculus from the introductory physics course taken by freshmen majors in physics, chemistry, and math. Our primary motivation was a gradual series of changes in the concurrent mathematics sequence which delayed the study of differential and integral calculus until long after they were needed in physics. Consequently, if we wished to use calculus notation or derivations, we had to spend valuable class time teaching mathematics, and this we were unwilling to do. So it was decided to place the science majors in the general non-calculus physics course, taking the opportunity to completely redesign this two-semester sequence.

One question we considered carefully was whether calculus actually was necessary to the teaching of physics at the introductory level. We finally concluded that it provided three basic contributions:

(a) It promotes a deeper appreciation that causal relations involve not only observable quantities but their rate(s) of change.

(b) Calculus provides a convenient way to do summations which occur when other than point masses are discussed.

(c) The instructor can assign more complex (and possibly more interesting) exercises to the student if he may use calculus to solve them.

We had already been making considerable use of digital computation in the calculus-physics course, and it was natural to think of using the computer to achieve these three objectives in the non-calculus course.

In the kinematics section of the course, we had been using an approach which placed heavy emphasis on functional relations and graphs from which the students got a fairly good appreciation of slopes (using $\Delta x/\Delta t$ notation). However, the "retrograde" relations (v--> x or a -->v) are usually derived in non-calculus texts through artifices involving the average velocity which leave the instructor and a small minority of the better students feeling uneasy. By using the techniques of numerical integration (we say "summation") these concepts are developed in a more natural way which also yields some important side benefits:

(1) Not only can this approach produce the formulas for the special case of constant acceleration, but it may be used to solve any dynamic problem for which the forces are known.

(2) Science majors who learn calculus later can make a simple change in notation (Δx --->dx, etc.) and immediately apply calculus techniques to physics problems.

(3) The inexactness of the numerical approach I actually consider an advantage, since it serves as a springboard to a discussion of accuracy and precision in physical measurement and calculation.

(4) The numeric summation techniques learned here can be directly applied to objectives (b) and (c) above formerly attained through the use of calculus.

A digital computer, of course, makes the numeric approach viable by removing the drudgery required for the many repeated steps needed for reasonable accuracy. When comparing the performance of our students with those in the years before this approach was instituted, we believe that the students who formerly would have taken the calculus course are doing almost as good a job at achieving objective (a) and the non-science or pre-med majors are doing much better in this area than in their former (non-calculus, non-computer) course.

A typical application for (b) might lie in thermodynamics or the study of rotation. Let us take the latter as an example. It is perfectly possible to get across the physical principles of rotation involving angular velocity, angular momentum and energy using a thin hoop and never to need calculus. However, students realize that most rotating bodies are not just thin hoops, and chafe at this restriction unless they are shown how real situations can be approached. The solution is to set up a numeric summation for a simple body (cylinder, rod, etc.) and show that $\vec{L}$ and K depend on mass and ω in the same way as the hoop, but also involve a geometry dependent factor. The students are then much more willing to accept a given table of moments of Inertia, once they see how they can be computed.

Very few problems in dynamics are ever solved directly from Newton's laws in a non-calculus course except for those involving constant linear acceleration or uniform circular motion. Objective (c) may be met by using numeric summation and a computer to investigate such interesting examples as a projectile with air resistance, a rocket (including relativistic effects if desired!), simple harmonic oscillators and even large-amplitude pendulums. (While most of the examples quoted so far have been in mechanics, it is just as easy to think of examples in other areas of the introductory course.)

Available Equipment

The first year in which the revised course was given, we were limited to batch processing using FORTRAN, and because of the attendant difficulties, only assigned eight computer problems in two semesters. In the past two years, we have acquired a small computer with a convenient interpretive procedure-oriented language (FOCAL) which we treat almost like a single time-shared terminal. By having the 100 students in the course work in pairs, the partnerships can use from one-half to one hour of terminal time each week.

At the start of the year, the students are given a booklet explaining the elementary features of the FOCAL language and containing "hands-on" examples to try. With no more than one terminal-hour's experience, the typical student partnership can use the system with confidence as a "super desk-calculator". No class time is used to discuss the computer up to this point.

In the kinematics section, the concept of a stored program is introduced in class, the students are given two "canned" programs, one to solve a tedious calculation problem given in the text which involves logarithms and another which performs a simple Euler summation to find the position of a body when its acceleration is known. About one-half hour of class time is devoted to explaining these programs, and the students then use them for a homework assignment. Another booklet is then handed out discussing some of the advanced features of the FOCAL language, including subroutines, looping and subscripted variables, and students are encouraged to modify the "canned programs" to make them more useful, such as inserting the value of Δt during the run, printing out only every 10th step permitting smaller step sizes for the same run time, etc. About two-thirds of the students do take this opportunity to write their own routines.

Later, after the concept of force has been introduced, "canned" integrators using the Leapfrog method are furnished for both one- and two-dimensional problems. All that needs to be specified is the force function, the initial conditions and the integration and printout steps. Another half-hour of class time is used to explain the Leapfrog method and to discuss computer usage in general. After this a fairly standard procedure is followed whenever a computer-usable problem is assigned. A "canned program" is provided to perform the basic computations, and a description of the program, instructions for using it, and suggestions for modifying the program and expanding the investigation are passed out to the students. Extra credit is given those who go beyond the bare minimum necessary to solve the assigned problem.

It is interesting to compare the numbers of students who are satisfied with using "canned" programs to those who use the computer in more imaginative ways:

Students who use the "canned" programs	95%
Students who use the computer in lab or as a calculating machine	85%
Students who modify the "canned" programs	50%
Students who go beyond the minimum assignment for computer problems	50%
Students who write their own programs	30-40%
Students who become really proficient in use of the computer and write programs to add to the library	10%
Students who tend to become "computer bums"	3%

The computer is turned on in the morning when anyone arrives and is turned off in the evening by the janitor when he leaves, and runs unattended most of the day. Instructions are left by the console on how to reload the interpreter in case the previous user has fiddled with the memory, a matter of no more than 40 seconds. We were rather surprised at the low number of SOS calls after the first week until we discovered that the students who caught on faster were teaching the others themselves and not bothering the instructors.

Summary

The instructors are happy with the new course and feel that the use of the computer has more than compensated for the deletion of the calculus requirement for science majors. Student comments in general seem favorable. About one-third requested more computer work, about one-third of the students thought there should be less use made of the computer, and the remaining third thought that the amount of computer exercises were "about right". It was not possible to have control sections, so I cannot include here the statistics beloved of learning researchers.

In the appendices are given a list of the computer assignments made during the two-semester course, and more detailed descriptions of several of these assignments showing the differences between the calculus and computer approaches.

About one hour of class time was used to explain the concept of numeric summation, and a total of only one more hour was used to discuss programming concepts and machine operation, the students learning the rest by trial and error at the terminal or from the instruction books. It is felt that this is much less than the time spent "teaching" calculus during the previous physics courses, so that we actually gained some class time to discuss additional physics topics.

APPENDIX I

DISCUSSION OF NUMERIC INTEGRATION TECHNIQUES

Euler techniques are always taught first because of their simplicity, using the algorithm

$$x_{new} = x_{old} + v\,\Delta t$$

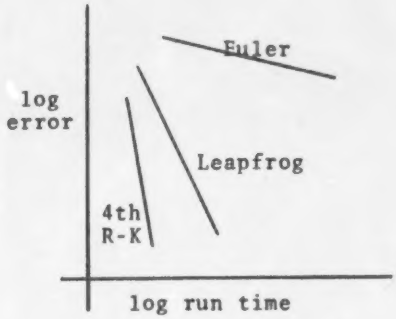
$$v_{new} = v_{old} + a\,\Delta t$$

Stepping either x or v first is equivalent to using the velocity at the beginning or end of the interval as an approximation to the average velocity during the interval. This leads naturally to using either the average of the initial and final velocities to step x (the Euler-Gauss technique) or using instead the velocity halfway through the interval (the Leapfrog method) according to the algorithm

$$x_{new} = x_{old} + v_{1/2}\,\Delta t$$

$$v_{3/2} = v_{1/2} + a_{(x_{new})}\,\Delta t$$

When using a numeric integrator, one is not concerned with its accuracy alone, but a combination of good accuracy with small calculation time, which we may call its "efficiency". Theoretical considerations suggest that the efficiency may best be measured by using an integrator on well known trial functions and plotting the error in the final result versus the total run time on a computer using a log-log graph. Most trial functions corresponding to problems one wishes to assign in introductory physics produce a graph like that shown below.



The lower left region of the graph describes integrators with high efficiencies. The typical order of efficiency for five common integrators is:

1. 4th order Runge-Kutta
2. Leapfrog
3. 3rd order Runge-Kutta
4. Euler-Gauss
5. Euler

Because it is much easier to teach and understand, the Leapfrog Integrator is used as the standard "canned" program furnished to students. Runge-Kutta programs are provided students in more advanced courses.

APPENDIX II

DISCUSSION OF SAMPLE ASSIGNMENTS

1. Harmonic Oscillator and Pendulum. Students are required to produce a graph of x-vs-t for a simple harmonic oscillator with specified initial conditions. Since they have already found that the accuracy varies with t, the choice of time step is left up to them. Two runs are required with different spring constants to see how that affects the problem. For extra credit, students may investigate how m affects the motion, or how the motion is changed when a pendulum force is used instead of a linear restoring force. The variation of period with amplitude of a pendulum has already been investigated in the laboratory, and the students may use the computer program to verify their results.

2. Rocket. The canned integrators are arranged so that all a student must do is insert an expression for the force function in a subroutine, then furnish the initial conditions and the timestep. To accommodate rocket problems, the student merely writes one more expression for the mass and places it immediately after the force statement.

3. Lunar Lander. Students get three chances to land a rocket on the Moon from a hovering position. The problem is complicated enough that there is little chance of success unless some preliminary analysis is done using the formulas for constant acceleration. For extra credit, students may try other strategies (non-constant acceleration) to attempt to optimize the fuel usage. (This problem follows a suggestion by H. Weinstock.)

4. Satellite Orbit. Students must find the correct initial velocity for a circular satellite orbit and a transfer orbit between two altitudes. Again pre-calculation reduces the number of trials. For extra credit, the students are urged to find the escape velocity and to experiment with different initial velocity directions.

5. Thermodynamic Equilibrium. A system of 100 "molecules" is displayed whose energy is quantized to the values 0, 1, 2, 3.... All are started with the same initial energy (say 2). As time goes on, random transfers of energy take place between the molecules. The student may stop the process at any time and request a plot of the number of molecules with various energies vs the energy, or look at the molecular system itself. In less than 5 minutes, the student can gather enough data to verify that the energy distribution approaches an equilibrium value which is exponential. Students are asked to choose another value for the starting energy to see if this will affect the final distribution or the rate at which equilibrium is approached. For extra credit another program is available which uses a quasi-dialog to guide the student to find all possible microstates of a system of n oscillators (typically n=3 to 5), and from these use the principles of probability to predict the equilibrium distribution.

6. Second Law of Thermodynamics.

The 100-particle system of program 5 is divided into two 50-particle sub-systems which are allowed to interact after giving them different initial energies (temperatures). It is verified that after the systems reach equilibrium, the hot system cools off and the initially cool system heats up. The energy and entropy changes can be traced. For extra credit, the possible states of a smaller system are predicted (same program used for extra credit in 5) and it is verified that it is more probable for the hot system to cool off. If this is done for two systems with different n, it can be seen that the probability increases dramatically as n increases.

7. The Electric Field Game. The student can request either the electric field vector or the electrostatic potential at any point in a rectangular domain. From the values given him by the program (which is simulating an electric meter) he must deduce the positions of any charge(s) within the domain and guess at their value. The program judges his guesses by their accuracy and the number of measurements he had to make. Several charge distributions are available for extra credit which are progressively more difficult to solve. This program has been used with a junior class in Electromagnetic theory also.

8. Lens Design. This program traces the path of light rays through a "real" thick lens (two surface). The student is asked to design a lens of specified aperture and focal length using spherical lens surfaces. He is judged by the percentage of rays which pass through a focal circle. Diffraction is not included, but the wavelength may be changed for extra credit to check on chromatism.

9. *Compton Scattering*. Students were asked to compute the wavelength shift of a light ray scattered from an electron by hand for a scattering angle of 180°. This program is used to calculate the wavelength shift (plus other descriptors) at any angle. Students are asked to produce a graph of $\Delta\lambda$ vs θ. For extra credit, they can investigate the effect of initial wavelength and mass of the scattered particle on the shape of the graph.

10. *Relativistic Rocket*. The velocity vs time (both lab and proper) of a rocket with constant thrust and mass (it scoops up hydrogen from interstellar space for its reaction mass and fuel!) is calculated. It is shown that the lab velocity approaches a limit of c, but that the proper velocity increases beyond bound. For extra credit, the students try to find the time after the rocket's departure when a light beam will not catch up to it.

11. *Energy Eigenvalues*. Everybody does this for a square well. For extra credit students may try a harmonic-oscillator well. We are trying at present to find a good way to present a nuclear well which will exhibit alpha decay.

12. *Nuclear Decay Chain*. An accelerator produces parent nuclei at a constant rate. The student may adjust the lifetimes of two daughter nuclei to see how this affects their number as a function of time. Turning the accelerator off will produce a simulation of a natural radioactive decay chain.

13. *Nuclear Reactor Simulator*. Unlike the other programs which had to have a specific pedagogical value, this one was just thrown in as an interesting toy at the end of the year. Students control the absorptivity of a nuclear reactor, and attempt to bring it up to a specific power level within a given period of time by withdrawing the control rods. Realistic "scrams" and poisoning are included, so some care is necessary for success. For extra credit students may remove the delayed neutron groups and see that this small minority of the neutrons is what makes a reactor stable.

APPENDIX III

A LIST OF INSTRUCTIONS AND ASSIGNMENT SHEETS GIVEN TO STUDENTS

1. Instructions on use of computer as "super-calculator".
2. Math exercises using computer functions.
3. Instructions of stored programs and programming techniques.
4. Vertical projectile.
5. Vertical projectile with air resistance.
6. Harmonic oscillator and pendulum. (#1 above)
7. Rocket. (#2 ")
8. Lunar lander. (Rocket with variable thrust.) (#3 ")
9. Satellite orbit. (#4 ")
10. Two-dimensional collision.
11. Summation of rotational energy and momentum.
12. Thermodynamic Equilibrium distribution. (#5 ")
13. Heat transfer and the second law of thermodynamics. (#6 ")
14. The Electric Field Game. (#7 ")
15. Summation of Sine-Waves.
16. Interference and Diffraction Patterns.
17. Lens design. (#8 ")
18. Relativistic Collision (Compton Scattering). (#9 ")
19. Relativistic Rocket. (#10 ")
20. Eigenvalues of matter wave in a square well. (#11 ")
21. Nuclear decay chain. (#12 ")
22. Nuclear reactor simulator. (#13 ")

A PHYSICS COURSE BASED ON A SET OF PROBLEMS SOLVED BY COMPUTER

Robert Ehrlich
State University College
New Paltz, New York

In the Spring 1971 semester, the Physics Department at the State University College at New Paltz is offering a new course dealing with computer applications in physics. This three credit course, which is open to all students who have completed one semester of General Physics, is currently being offered under the catalog listing: "Selected Topics in Physics". The course consists of a two to three week introduction to computers and FORTRAN programming, followed by a series of ten weekly problems covering topics in the areas of electricity, magnetism, waves and modern physics. The choice of topics roughly corresponds to the subject matter in the second semester of our two semester General Physics course, which many of the students enrolled in this course are taking concurrently. A set of notes, written by the author, containing an introduction to FORTRAN and a discussion of each of the ten problems, serves as the text for this course. Most of the problems in the course are, in essence, simulated laboratory experiments; a number were patterned after actual experiments in our General Physics course, for example, plotting of equipotential surfaces, calculation of the charge on a discharging capacitor, and plotting the interference patterns for two point sources of waves. The notes covering each problem include a discussion of the theory and a well-documented FORTRAN IV computer program, written for an 8,000 word IBM 1130. For each weekly problem, a student is expected to run the program supplying data of his own choosing, make a number of suggested modifications, and discuss the results in a paper analogous to a lab report, which is handed in each week, together with program listings and output. There are no examinations in the course. A student's grade is partly determined on the basis of the weekly reports, and partly on the basis of a term project of his own choosing. The term project is expected to be similar in style to the weekly problems, but it would require more initiative on the part of the student.

The objectives of the course are:

1. To allow the student to acquire a working knowledge of computer programming, and in particular, to become familiar with the FORTRAN language, through its use in the solution of a variety of physical problems.

2. To give the student a better understanding of certain physical phenomena, by allowing him to see, with the aid of a computer, how the nature of the phenomena depends on the numerical values of a number of parameters, which he is free to specify.

3. To show the student how a number of mathematical techniques suitable for solving physical problems, such as numerical integration, step-wise solution of differential equations, Fourier Analysis, and Monte Carlo methods, can be implemented on the computer.

4. To show the student how to generate graphic displays on the computer, permitting results to be presented in a more meaningful and immediately transparent manner, than is possible with a strictly numerical tabxlation of results.

The only plotting device available is the printer on the IBM 1130. Nevertheless, in many cases, this is more than adequate. (See, for example, Figure 1 of the appendix.)

The set of ten problems is listed in Table 1. One of the problems is discussed in detail in Appendix I, which is copied from the set of notes that comprises the text for the course.

At the time of this writing, the course has not yet begun; making any attempt at an evaluation impossible at this time. One foreseeable problem concerns the non-homogeneity of the group of approximately ten students enrolled in the course. There is a fairly wide range both in physics background and also in computer experience among the students in the class. It is hoped that by allowing for a large amount of student initiative, more advanced students will be encouraged to proceed at their own rate, and possibly substitute problems of their own choosing for some of the existing ones. Similarly, while FORTRAN is the "official" language for the course, students wishing to use some other language will be free to do so.

APPENDIX I: Interference Patterns Produced by Two Point Sources of Waves

Waves coming from two or more coherent sources will produce an interference pattern. An interference pattern arises because the resulting intensity from a number of coherent sources is found by adding the amplitudes of the waves reaching a given point, and squaring the result to find the resultant intensity. That is, we add amplitudes, (which may be positive or negative), rather than adding intensities, (which cannot be negative, since intensity is proportional to the square of the amplitude). Thus, it is possible for waves from two sources to constructively interfere at some points, where the wave amplitudes have the same sign, and destructively interfere at other points, where the wave amplitudes have opposite signs. This can be contrasted with the case of two incoherent sources, in which we add intensities rather than amplitudes, and for which no interference pattern can be observed. (Why?)

Intensity for two coherent sources: $I = (A_1 + A_2)^2$

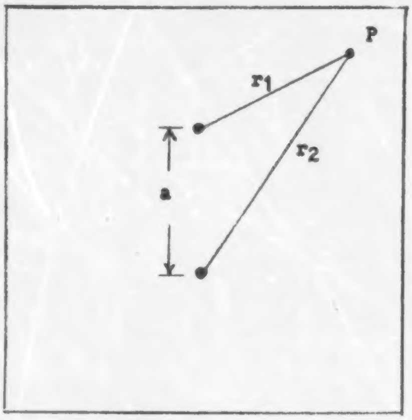
Intensity for two incoherent sources: $I = A_1^2 + A_2^2$

For a spherical wave from an isotropic point source, the wave amplitude a distance r from the source, may be written:

$$A = (S/r) \sin \frac{2\pi}{\lambda} (r - ct)$$

In this equation, S is determined by the strength of the source, λ is the wavelength and c is the wave speed.

Given two sources, separated by a distance a, which emit waves that are initially in phase, the amplitudes reaching point P, shown in the figure, at a given time, say $t = 0$, can be written:



$$A_1 = (S_1/r_1) \sin 2\pi r_1/\lambda_1$$

$$A_2 = (S_2/r_2) \sin 2\pi r_2/\lambda_2$$

Once numerical values have been supplied on a data card for the parameters; S1, S_2, λ_1, λ_2 and a, a computer can be used to find the resulting intensity: $I = (A_1 + A_2)^2$ at a large number of points in space, and generate an interference pattern. Instead of printing the resulting intensity at an array of points, the computer represents the intensity at each point by a symbol, such that high intensity points are represented by symbols that look dark on the page, and low intensity points by symbols that look light. The resultant patterns, shown in Figure 1, are photographs of the printed output obtained for four different sets of parameters. The ability to print symbols having a wide range from light to dark, depends on the capability of over-printing lines on the printer. To print lines of characters on top of each other on the printer for the IBM 1130 and many other computers, all that is required, is to use a plus sign as a control character, i.e., the first character on a line to be printed. In this case, the printer will print a line of characters without advancing to the next line, as is usually the case.

The computer program which generated the plots shown in Figure 1 works in the following way. For each of 61 rows of points, the program calculates the resulting intensity of waves from two point sources, reaching each of 101 points within a row. It then uses some function of the intensity at each point to obtain a single digit from 1 to 9. The function actually used involves a logarithm to make it slowly varying, but is quite arbitrary otherwise. The "intensity digits" calculated for each of the 101 points within a row are stored in the array JE(K), K = 1...101. Next, the computer prints out a row of symbols, such that for each position where the intensity digit is 1 or less, a blank is printed, and for each position where the digit is two or greater, a certain character, actually an "I", is printed. Then, the computer subtracts one from each intensity digit in the row and repeats the procedure, printing a blank if the intensity digit is one or less, and printing another character, this time a "-", if the digit is greater than or equal to two. This procedure is carried out a total of six times, so that at any given position on a line, anything from a blank to six symbols on top of each other may appear, depending on the intensity at that point.

Run the computer program for this problem, using various values of the parameters. Modify the program to handle the case of two incoherent sources and make any other changes that occur to you.

TABLE I: Physics Problems Studied Using a Computer

1. Electric Field of a Point Charge. In order to illustrate the fundamentals of FORTRAN, the electric field of a point charge is computed for a range of distances.
2. Equipotential Surfaces. The equipotential surfaces are plotted by computer, for the case of two point charges.
3. Potential of a Line of Charge. The finite line of charge is approximated by a finite number of point charges, and the electrical potential is computed for a range of distances.
4. RC Circuits. The charge remaining on a discharging capacitor in an RC circuit is computed and plotted. Both the exact result, and an approximation found from a step-wise solution of the differential equation are displayed.
4. Random Phenomena. As an illustration of the use of random numbers in a calculation, random numbers are generated by computer in a calculation of pi. The problem of the "random walk" is also treated.
6. RLC Circuits. The current is computed and plotted as a function of time, for the case of free oscillations in an RLC series circuit.
7. Fourier Analysis. A square wave is synthesized using arbitrary numbers of sine waves, and the result is tabxlated and plotted.
8. Interference. The interference pattern produced by two point sources of waves is displayed.
9. Chain Reaction. A fission chain reaction is simulated using a Monte Carlo technique, and the critical mass of a rectangular slab of uranium is computed.
10. Bubble Chamber Analysis. Measurements made on a number of bubble chamber photographs are analyzed, and the mass of an undetected neutral particle is computed.

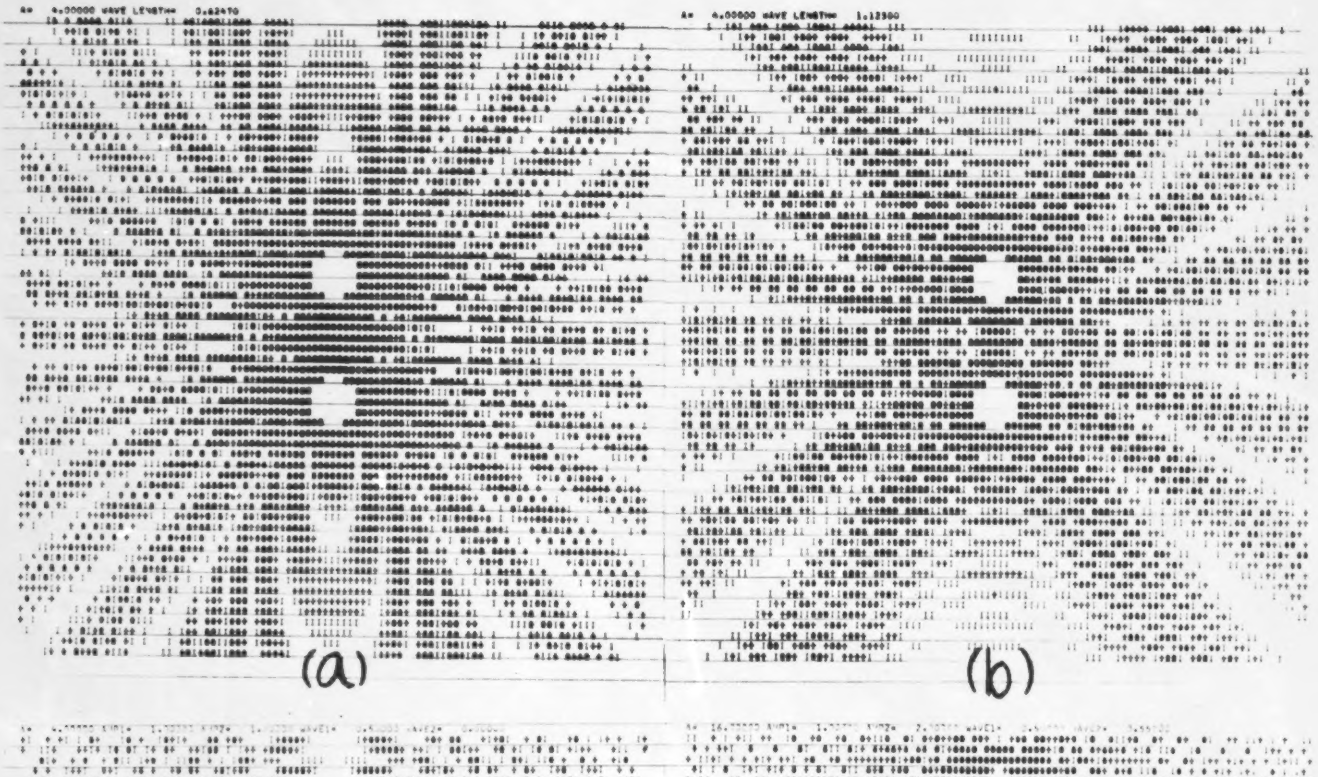

A= 4.00000 WAVE LENGTH= 0.62470
A= 4.00000 WAVE LENGTH= 1.12300
(a)
(b)
(c)
(d)

FIGURE 1

COMPUTER INVESTIGATION OF NORMAL MODES OF VIBRATION FOR COUPLED OSCILLATORS

Herbert D. Peckham
Gavilan College
Gilroy, California

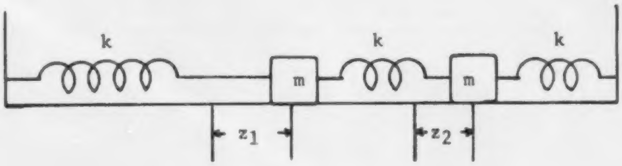
Abstract

A computer program is described which has been used with sophomore science and engineering students to investigate the highest and lowest normal modes of vibration for coupled oscillators with N degrees of freedom. The development of the theoretical model and the numerical solution algorithm is given. The student reaction and the instructor's evaluation of the results are presented.

Theoretical Model

Most introductory physics texts take up the question of the behavior of coupled oscillators. Generally, the types discussed are mechanical mass-spring systems or LC electrical circuits. The usual technique is to discuss in detail the characteristics of systems with two degrees of freedom and to terminate the treatment before becoming involved with systems with more degrees of freedom. In this paper an experiment will be discussed which used the computer to investigate the normal modes of vibration for coupled oscillators with N degrees of freedom.

To develop the theoretical model, consider a mechanical mass-spring system consisting of two masses, each of mass m, sliding on a frictionless horizontal surface, coupled to each other and to fixed end supports by identical springs with spring constants k.



Let z_1 and z_2 denote the displacements of masses 1 and 2 respectively from the equilibrium positions. For the purpose of clarity we assume that z_1 is greater than z_2. If we apply Newton's 2nd law to both masses we obtain the following equations:

$$m(d^2z_1/dt^2) = -kz_1 - k(z_1 - z_2) = -2kz_1 + kz_2 \quad ,$$
$$m(d^2z_2/dt^2) = +k(z_1 - z_2) - kz_2 = +kz_1 - 2kz_2 \quad .$$

Thus, the equations of motion for the system are:

$$(d^2z_1/dt^2) = -2(k/m)z_1 + (k/m)z_2 \quad ,$$
$$(d^2z_2/dt^2) = +(k/m)z_1 - 2(k/m)z_2 \quad .$$

For normal modes of vibration, we can assume the solutions

$$z_1 = c_1e^{i\omega t} \; , \; z_2 = c_2e^{i\omega t}$$

where c_1, c_2, and ω are to be determined. Furthermore, we note that $(k/m) = \omega_o^2$, the square of the fundamental frequency of vibration of a single, undamped mass-spring harmonic oscillator. If the assumed solutions and substitutions above are made in the equations of motion, and the results converted to matrix form we obtain

$$\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = (\omega/\omega_o)^2 \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$$

which is the standard eigenvalue problem

$$AX = \lambda_n X.$$

The eigenvalues of the matrix A give us the normal frequencies of vibration in terms of the fundamental frequency of vibration. Thus, one of the interesting problems in coupled oscillators (the normal frequencies of vibration) can be solved if the eigenvalues of the associated matrix can be found.

If the analysis above is repeated for a coupled system with three degrees of freedom (three masses coupled together and to fixed end supports) the corresponding matrix A is

$$\begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}$$

At this point, we can generalize the results to a system with N degrees of freedom. The matrix A will be N x N, will have 2's on the main diagonal, and -1's on the super and subdiagonals. All other elements of the matric will be 0. The eigenvalues of the matrix A contain all the information concerning the normal frequencies of oscillation of the system.

Computer Solution

Most often, the method utilized to find the eigenvalues of a matrix A is to set

$$\det(A - \lambda I) = 0 \quad ,$$

which is equivalent to the solution of a polynomial in for its roots. The problem is that as N increases, the practical difficulties involved in identification of the roots become severe. Instead, we will use an iterative numerical scheme to find the largest and smallest eigenvalues and associated eigenvectors. The proof of the method is given by Arden and Astill[1]. The numerical scheme and BASIC program to carry out the algorithm has been discussed by the author(2). The algorithm is as follows:

$$gX_{i+1} = AX_i \quad .$$

An arbitrary vector (in the program the vector with all components equal to 1 is chosen) is used to start the process. After multiplying by A, g is factored out of the result such that the last element in X_{i+1} is equal to 1. This vector is used in the next stage of the iterative process. As the process goes on, g converges on the largest eigenvalue of A, and X converges to the associated eigenvector.

To find the smallest eigenvalue and associated eigenvector we use the algorithm

$$hX_{i+1} = A^{-1}X_i \quad .$$

The process is identical to that described above except that the factor h converges to the reciprocal of the smallest eigenvalue as X converges to the associated eigenvector.

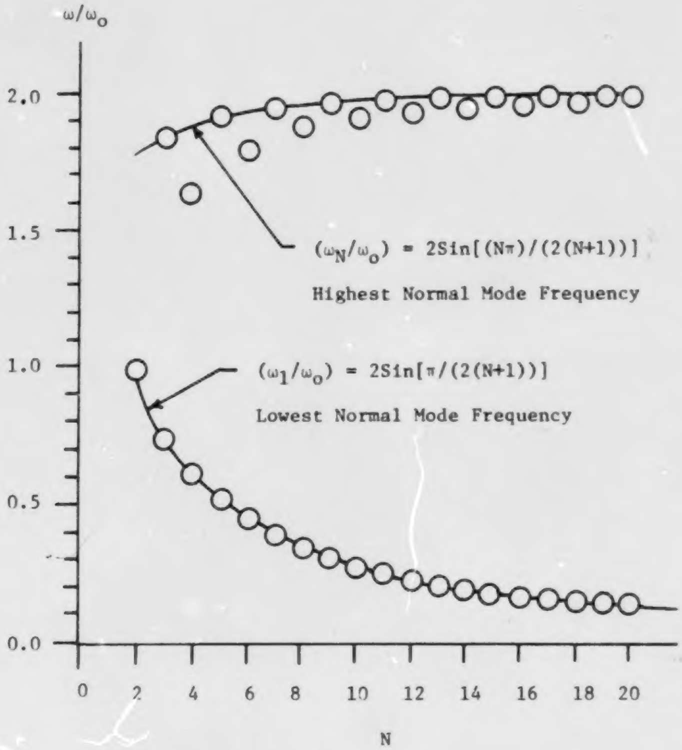
A BASIC language program to carry out these algorithms is given in Appendix A. The program can handle up to 10 degrees of freedom as is. If greater numbers of degrees of freedom are desired, the DIMENSION statement in line 102 must be re-written. The number in the DATA statement in line 800 is the number of degrees of freedom in the problem. This is followed by

DATA statements containing the rows of the matrix A. Appendix B contains the print out of the program for N = 3.

The details of the structure of the program are discussed in the reference already cited(2). The students who took part in the experiment were not required to write the program. They were given the theoretical development above, and instructions for use of the program.

Student Investigation

The students were asked to discover what happens to (ω_{max}/ω_o) and (ω_{min}/ω_o) as the number of degrees of freedom increases. The program was run for N = 2, 3, 4, ..., 20. In each case the desired values (ω/ω_o) are the square roots of the eigenvalues. The results are shown in the diagram below.



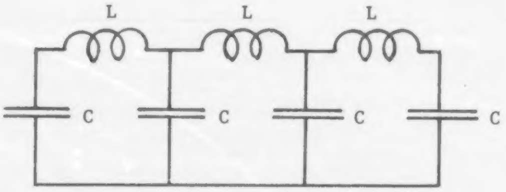
The circles are the computer results. The solid lines are the theoretical solutions obtained from an advanced reference(3). The results immediately posed a dilemma. The computer results for the lowest normal mode frequencies, and for the highest normal mode frequencies for odd degrees of freedom are in exact agreement with the theoretical results. However, for the even degrees of freedom and the highest normal mode frequencies, there is serious disagreement.

After much study of the computer results the students discovered that for the cases of disagreement, the program had converged on the next highest eigenvalue instead of the highest. This problem was solved by choosing an initial vector in the program with the first component equal to 1 and all others equal to zero. With this change, the computer results and the theoretical values are in exact agreement.

However, the solution of the problem raised another. The proof for the numerical method admits an arbitrary vector for the initial choice, and proves that the process always converges to the highest eigenvalue. Clearly this didn't happen in the student example. The students were intrigued by the new problem which has not been resolved.

The computer results show several fundamental ideas very clearly. As N increases, (ω_{max}/ω_o) approaches 2. But this is the high frequency cut off for signals on such a coupled system (twice the fundamental frequency of oscillation). Also, (ω_{min}/ω_o) approaches 0 as N increases. This is the low frequency cut off. The results therefore show that in the limit as N increases, the system comprises a bandpass system which passes signals between 0 and twice the fundamental frequency of oscillation.

As a laboratory adjunct to the investigation, an electrical analog to the coupled oscillator system was constructed. By writing the differential equations for a simple mass-spring system, and a LC oscillator and noting the relationships between the coefficients it was fairly easy to show that a system with N = 3 (as an example) has the electrical analog below.



Several electrical systems were constructed, excited with a signal generator, and normal frequencies of oscillation observed on an oscilloscope connected across the circuit. Considering the error arising from tolerances of the components used in the circuits, the experimental agreement with computer predictions was quite good.

Evaluation of the Experiment

The student reaction to this kind of experiment was enthusiastic. When it started, no one (including the author) knew what the results would be. The students generally enjoyed extending a text presentation with the aid of the computer. They were particularly excited by the discovery that the initial choice of vector in the program does make a difference. One student observed that most of the components of A when N is large are 0. He is at present rewriting the program to close pack the matrix and permit the computer to handle matrices up to N = 120. Considering the computer upon which the program was run (Hewlett Packard 2114A with 8K memory) this is quite an achievement and indicative of what some students will do if given the chance.

From the instructor's view point the experiment was a resounding success. It gave an opportunity to introduce the concept of eigenvalues and eigenvectors in a simple context that was closely related to the current text material on waves. The experiment was certainly "non-cookbook" and was susceptible to laboratory investigation. Finally, the techniques utilized can be applied to numerous other physical systems with, it is believed, equal success.

In the broader sense this paper illustrates a type of computer usage. The computer is used to enhance ideas and develop concepts. Certainly more is involved here than simply using the computer as a high speed calculator. In the opinion of the author, this is the type of methodology which is likely to be the most valuable to the college with very limited computer resources.

REFERENCES

(1) Arden and Astill, Numerical Algorithms: Origins and Applications, Addison Wesley, 1970, page 201

(2) Peckham, Computers, BASIC, and Physics, Addison Wesley, 1971, page 181

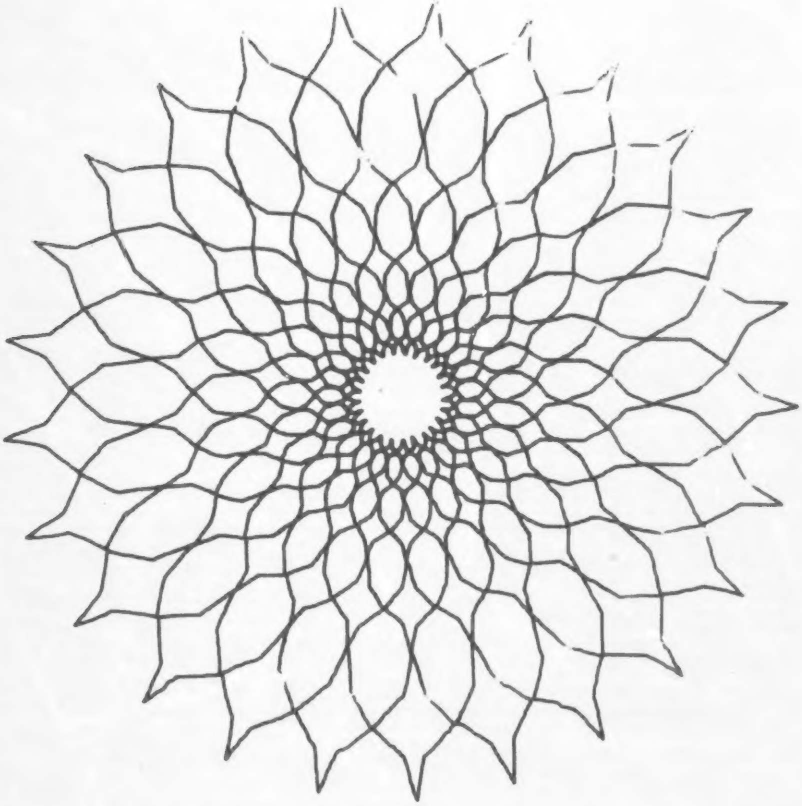
(3) Morse and Feshbach, Methods of Theoretical Physics, McGraw Hill, 1953, Vol. 1, page 134

APPENDIX A

```
LIST
100  REM EIGENVALUES AND EIGENVECTORS
102  DIM A[10,10],B[10,10],X[10,1],Y[10,1]
110  READ N
120  MAT  READ A[N,N]
130  LET S=K1=K2=K3=0
131  LET K=1
140  MAT X=CON[N,1]
141  MAT Y=ZER[N,1]
142  MAT B=ZER[N,N]
150  MAT Y=A*X
160  IF Y[N,1]=0 THEN 170
161  GOTO 180
170  LET Y[N,1]=1.00000E-03
180  FOR I=1 TO N-1
190  LET Y[I,1]=Y[I,1]/Y[N,1]
191  NEXT I
200  LET K3=K2
201  LET K2=K1
202  LET K1=K
203  LET K=Y[N,1]
204  LET Y[N,1]=1
210  IF ABS(K-K3)=0 THEN 220
215  MAT X=Y
216  GOTO 150
220  IF S=1 THEN 260
230  PRINT
231  PRINT "ESTIMATE OF LARGEST EIGENVALUE IS "K
232  PRINT "CORRESPONDING EIGENVECTOR IS"
233  FOR I=1 TO N
234  PRINT Y[I,1],
235  NEXT I
240  MAT B=INV(A)
241  MAT A=B
250  LET S=1
251  GOTO 140
260  PRINT
261  PRINT
262  PRINT "ESTIMATE OF SMALLEST EIGENVALUE IS "1/K
263  PRINT "CORRESPONDING EIGENVECTOR IS"
264  FOR I=1 TO N
265  PRINT Y[I,1],
266  NEXT I
800  DATA 3
801  DATA 2,-1,0
802  DATA -1,2,-1
803  DATA 0,-1,2
999  END

READY
```

APPENDIX B

```
RUN

ESTIMATE OF LARGEST EIGENVALUE IS  3.41421
CORRESPONDING EIGENVECTOR IS
 1              -1.41421         1

ESTIMATE OF SMALLEST EIGENVALUE IS  .585786
CORRESPONDING EIGENVECTOR IS
 1.              1.41421         1
READY
```



William Carney

INTERACTIVE SIMULATION-GAMES IN ECONOMICS

John A. Naylor and William M. Baird
The College of Wooster
Wooster, Ohio

I. COMPUTER FACILITIES

The College of Wooster has two types of computer facilities available to it. A Control Data 6400 computer is available by means of a remote terminal batch-processing arrangement with the Battelle Memorial Institute in Columbus, Ohio. The College has found this to be a very desirable arrangement, both in terms of cost to the College and in terms of the computer power available to us. In addition to the Battelle hook-up, the college also has teletype-terminal access to the BASIC and FORTRAN-dedicated computers of Leasco Data, Inc., whose home offices are in Washington, D. C. As most of this audience is aware, this type of installation is ideal for running jobs which require a relatively small amount of data, and for the development of interactive programs of various types.

II. WHY INTERACTIVE ECONOMIC GAMES

We believe that the flexibility and power of interactive computer models enable us to better achieve several ends of traditional instruction. The primary benefits of well-designed interactive models seem to be

1) Increase in student interest in the subject matter.

2) Development of student appreciation of the complexity of policy makers' problems.

3) Utilization of a greater number of and more sophisticated economic models.

4) Improvement in ability to illustrate the practical value of theory and statistics in dealing with complex problems.

5) Provide a stimulus to the faculty to be more precise in specifying economic relationships; programming requires the development of precise functional forms and parameter specification which causes somewhat more deliberation on the part of the instructor.

6) Greater repetition and variety in expressing the same basic educational messages are accomplished.

III. POLICY SIMULATION GAMES FOR MACROECONOMICS

A. For use in courses such as principles of economics, macroeconomics, monetary economics, and public finance a variety of economic models have been programmed. These are similar in nature to the models of Attiyeh and Bergstrom. To this date these have been designed for the teletype I/0 model on a time sharing system though we have plans to convert a number of these to a batchprocessing system with both teletype and card/line printer I/O. The macro oriented programs, apart from differences in complexity, fall into two distinct categories. There are the nongaming or nonpolicy oriented programs designed to illustrate certain fundamental processes such as the demand deposit expansion process or the multipliers. In these programs the student merely specifies parameters and observes the level and time pattern of response. These programs are pedagogically useful only for those students with poor mathematical intuititon who thus need to have numerical demonstrations (e.g. that the greater the mpc the greater the multiplier) of functional relationships.

B. The second type of macroeconomic program is the interactive simulation program. It is this approach that we feel that has been most successful in stimulating students' interest and increasing their comprehension of the complexity of policy decisions and the interaction of various economic variables. First we shall discuss the nature of the computer model designed for use in the macrotheory courses and indicate the character of the models used in principles courses and finally note the variations deemed useful in the monetary economics course.

1. The purpose of the interactive simulation model is to place the student in a position in which he receives macroeconomic information analogous to, but simpler than, that received by, say, the FED or CEA. This information consists of both economic data generated by his own prior policy decisions, and revelations of exogenous shifts in parameters. In addition, various constraints on the use of policy instruments and specifications as to some short run policy targets are imposed.

2. The basic aggregate demand model used is roughly of the standard LM-IS type (with all due apologies to the monetarists) with term-structure and partially endogenous money supply features added. One actual model is given in Attachment A. Unemployment and price indices are determined by the relation of aggregate demand to potential supply which is a function of technology, labor, and capital; all of which vary over the game period. The first two grow at several preset rates while the capital stock varies with the investment demand in the macro model; hence the time path of potential output is peculiar to the policy mixes of each student running the game.

Since the model generates potential output based on capital stock values where $K_t = K_{t-1} + I_{t-1}$ and because it is desirable to enable the student to run the game at will and usually for as many quarters as he pleases, and because easily accessible records of each student's progress are desirable, the program used requires a separate memory unit for each student. This in turn calls for an access code for each student.

3. Before offering some specific examples of the types of parameter shifts and policy constraints involved, let's indicate how the game is integrated into the course material and supervised. After the students have been exposed to several weeks of aggregative models but before they have dealt with models with monetary sectors, the students are given a set of instructions as to how to operate the teletype in order to run the game and game ouput for several sample quarters (including the initial equilibrium values from which all students start). After a brief amount of discussion concerning the impact of monetary and fiscal variables, the students are permitted to run several quarters (there are 40 in all) of the game. At this juncture the students have not the theoretic framework to make judicious policy moves and usually institute perverse economic policies. This provides an excellent point of departure into two sector macro models because the students quickly realize that simple, static, one-sector models do not yield enough predictive power and having foundered their economies (in which they soon develop a distinct sense of identity) they welcome a theoretic framework that will enable them to make intelligent policy decisions.

As the course develops the students continue to run the game and are encouraged to relate the interactions they observe in the computer model to those developed in classroom. Occasionally, the instructor asks the class members to quantitatively express their policy objectives (i.e. indicate the exact inflation, and unemployment rates they will accept at the various output levels) or the instructor himself specifies the priority ordering to see if the students can develop the proper policy mix.

4. Periodically (about once every four quarters) as the student runs the game, he is informed of either parameter changes or restrictions on his policies. These are some typical "bulletins" to the policy-maker:

a. THIS IS THE QUARTER BEFORE THE NEXT IMPORTANT ELECTIONS. POLITICAL REALITIES REQUIRE THAT YOU BALANCE YOUR BUDGET OR FACE POLITICAL DEFEAT.
b. AN EFFECTIVE CAMPAIGN BY THE ENVIRONMENTALISTS IS LEADING TO A DECLINE OF .05 IN THE MPC.
c. THE DIRECTORS OF THE FEDERAL RESERVE BANKS HAVE JUST ANNOUNCED A ONE PERCENTAGE POINT (100 BASIS POINTS) DECLINE IN THE DISCOUNT RATE.
d. THE LONG-EXPECTED ASCENDANCY OF THE QUANTITY SCHOOL HAS ARRIVED. YOU NO LONGER CAN CONTROL NONBORROWED RESERVES. THEY WILL GROW AT A CONSTANT RATE HENCEFORTH.

See Attachment B for one structure of exogenous changes or restrictions built into the game. In addition, each policy action is rated by the program through a social utility index ($U=U(C,I,G,\dot{P},U)$) based on the level and mix of GNP, the rate of price increases and unemployment; the current and cumulative value of this index is output each quarter to help motivate the student. Also, if the student should employ an especially accurate or disastrous policy he receives an appropriate message designed to sustain his interest in the game, e.g.

a. LOOK AT THAT INCOME GAP AND UNEMPLOYMENT! ARE YOU A HERBERT HOOVER FAN?
b. THAT WAS AN EXTREMELY ACCURATE POLICY ADJUSTMENT. LORD KEYNES WOULD BE PROUD OF YOU.

At the principles level, we have developed models with the same general objectives but of much greater simplicity. We find that the average principles student is no more able to organize and assimilate the output of, say, a twelve equation model than he is able to comprehend fully the interactions of the real economy. To avoid frustrating him the problems must be simplified but not trivial (to the student). Principles students seem to exhibit less of a capacity (or desire) to adopt the macro policymaker role and clearly have greater problems in organizing the data generated. The problems that they encounter however, seem to enable us to better determine where our instruction is failing. It appears to us that the comprehension of macro principles as evidenced by passing exams, is more apparent than real and is probably a manifestation of rote learning behavior. If the students really understood their macro material, they would be able to organize the game-generated information, order their goals and specify the policy variables. Evaluation of their performance in policy simulations designed to stimulate their interest thus provides further evidence of their understanding macro principles.

IV. THE COLLECTIVE BARGAINING GAME

The economics curriculum at The College of Wooster includes a single course in labor economics. The content of the course includes the economics of the labor market, a brief summary of labor history and labor legislation, and a more extensive treatment of collective bargaining. Of the various topics included under the general heading "collective bargaining," the process of contract negotiation is usually the most difficult to deal with in the classroom. One may talk, for instance, about the various factors which the respective negotiators will consider in formulating their positions, and about the relative weights which they may assign to these factors. This procedure, however, is quite unsatisfactory. Each negotiation is indeed a separate phenomenon and the factors which are important in one negotiation may be entirely absent in another. In addition, the particular negotiating strategies also vary widely from one negotiation to another, thus making it difficult to generalize about this aspect of the negotiation process.

Structure of the Game

The structure of the game is very simple. One student (or team of students) plays the role of management negotiator and another takes the role of labor negotiator. The negotiators begin the game by entering two numbers into the computer, which then uses these numbers to generate data about the firm and the economy, and which the negotiators can use to help formulate their wage offers and demands (see attachment C). For example, this background information includes (1) the firm's demand for labor function; (2) the percentage change in the firm's profits since the previous negotiation; (3) the change in labor productivity since the last negotiation; (4) the current wage rate; (5) the current level of employment in the firm; (6) the size of the union strike fund; (7) the change in the consumer price index since the last negotiation; (8) the average hourly wage increase recently achieved by other labor unions; (9) the current rate of employment in the economy; and (10) the attitude of the current political administration toward the wage-price guideposts. The negotiators may choose to give this information a great deal of influence in forming their wage offers and demands, or they may choose to ignore all or any part of it. After this preliminary information has been printed at the terminal, the management negotiator moves away in order to allow the union negotiator to input certain confidential information to the computer. This information consists of (1) The union's attitude toward the employer; (2) the union's attitude toward the wage-price guideposts; (3) the actual wage goal which the union hopes to achieve; and (4) the length of strike necessary to convince the employer to agree to the union's terms.

The union negotiator then retires and the management team returns and inputs similar confidential information. Both negotiators then return to the terminal and input tne wage offer or demand they are making to the other team, the computer then calculates the bargaining attitude of each negotiating team. A negotiator's bargaining attitude may be defined as:

$$BA = \frac{\text{THE COST OF AGREEING WITH THE OPPONENT ON THE OPPONENT'S TERMS}}{\text{THE COST OF DISAGREEING WITH THE OPPONENT}}$$

For example, the employer's cost of disagreeing with the union on the union's terms would be the difference between the employer's wage objective and the union's wage demand, summed over the number of hours of work to be hired during the life of the contract. His cost of disagreeing would be the losses he would incur during a strike of the length he anticipates if he does not agree with the union's terms. The cost of disagreeing is an exponential function of the anticipated length of the strike.

If the bargaining attitude of both parties is calculated to be greater than one, a strike will occur. If the bargaining attitude of one of the parties is less than one, that party will accept the other's offer. If both bargaining attitudes are less than one, the union will accept the management's offer.

In the event a strike occurs the program loops back to the point where the union is called upon to input its confidential information, and the program will continue looping until one of the bargaining attitudes drops below a value of one. At the end of each round of negotiations their terminal prints out the losses to each side that have been incurred as a result of the strike so far. When the strike is settled, the terminal will print out the wage rate which has been negotiated, and each side's total losses due to the strike.

Evaluation of the Bargaining Game

At this time the bargaining game has been used only once by a class in labor economics. Therefore it is difficult to arrive at a comprehensive statement of the successes or failures of the programs, but some tentative generalizations may be cautiously set forth.

(1) Students have responded well to the game. Perhaps the greatest benefit derived from using the game has been that student interest in the area of collective bargaining has increased substantially. The game has provided a good take-off point for class discussions of collective bargaining, and a number of students have been motivated to do additional reading on this and related topics.

(2) Students have taken the game seriously and have become immersed in their roles of labor and management negotiators. They tend to be "hard-nosed" negotiators who are determined to obtain maximum concessions from the other side. In many cases they tend to ignore the cumulative costs of a strike in an attempt to maximize (or minimize) the negotiated wage rate.

(3) Although students have no good criteria against which to compare their feelings, they do express the attitude that they come away from the game with a better appreciation of the negotiation process. Because the game is intentionally long and tedious (if a strike occurs) the students become more aware of the frustrations and delays encountered in a long contract negotiation.

(4) The most frequently expressed complaint about the game is that the computer makes the critical decision of whether or not to agree to a proposed wage rate instead of the students themselves. This criticism is probably a valid one, and a future version of the game will merely provide a complete set of data which projects the anticipated costs and gains from pursuing a particular wage goal, and the critical decision will be left to the student negotiator.

Conclusions

In general the game appears to have been quite successful in achieving the limited goals set for it--if for no other reason than that it has significantly increased student interest in the subject matter. However, it is but a first step. Work is beginning on a much more complex simulation-game in which there will be multiple roles (stockholders, management negotiators, union negotiators, rank-and-file union members, public officials, etc.), and a much more complex decision matrix. Hopefully a report on this attempt can be made at some future conference such as this.

V. SUMMARY

Our brief experience with simulation games has been very gratifying. They stimulate student interest, better enable the instructor to discover areas of lack of understanding of economic relationships, create a better comprehension of the complexity of policy makers' problems, convince students that theory can be relevant and useful and, last but not least, they cause the instructor to think seriously about the nature and specification of the models if he wants to illustrate the presumed real world. There appears to be negligible negative reaction to the games once the student becomes familiar with the operation of the teletype. In fact, it has become necessary to program weekly limits on the number of rounds that can be played so that the students do not run too many rounds at once. To the novice the teletype is frequently no more than a complicated toy and many students initially tend to randomly type numbers rather than develop policy strategies.

NOTES

*Based on Cartter and Marshall, Labor Economics, R. D. Irwin, 1967.

ATTACHMENT A[1]

1. $Y = C + I + G + F$
2. $C = a + bY_d$
3. $I = cY_o - d(RL) + I_o$
4. $G = G^*$
5. $F = F_o$
6. $T = eY + T_o^*$
7. $M_d = fY - g(RS) - M_{do}$
8. $M_s = iW^* + j(RS) - k(RD^*)$
9. $RL = q(RS) - (1+q)(RN)$
10. $RN = (RN)$
11. $M_d = M_s$
12. $L_t = (1+g_1)L_{t-1}$
13. $K_t = K_{t-1}Q + g_2 I_{t-1}$
14. $A_t = (1+g_3)A_{t-1}$
15. $Q_t = A_t K_t^a L_t^{1-a}$
16. $I_o = m(SIN(t+1))$
17. $Y_d = Y-T$
18. $U = U(Y,Q)$
19. $P = P(U)$
20. $WEL = W(C,I,G,\dot{P},U)$

Y = net national product

C = consumption demand

I = net investment demand

G = government demand

F = net foreign demand

RL(S) = long-term (short) interest rate

RN = "normal" long-term rate

RD = discount rate

$M_{d(s)}$ = money demand (supply)

L = total labor supply

K = capital stock

A = technology index and scalar

Q = potential output

T = tax liabilities

Y = disposable income

U = percent unemployment

P = percent rate of inflation

WEL = policy makers' utility index

* potential policy control variable

[1] For a discussion of this IS-LM model see W.L. Smith, Macroeconomics, (Homewood: R.D. Irwin, 1970), Chapter 14.

Any variable without a time (t-1) subscript refers to the t-th period (t-0).

ATTACHMENT B

SAMPLE TIME STRUCTURE OF EXOGENOUS PARAMETER CHANGES AND POLICY RESTRAINTS

$t = 1$ Notification of periodicity in autonomous investment

$t = 4$ Exogenous increase in exports

$t = 7$ Exogenous decrease in m.p.c.

$t = 12$ Exogenous decrease in discount rate

$t = 21$ Requirement that the budget be balanced.

$t = 23$ Exogenous shift in autonomous investment

$t = 26$ Notification of increased rate of growth of labor supply

$t = 29$ Requirement that long-term rates be held below short-term rates (operation twist)

$t = 34$ Nonborrowed reserves now grow at a fixed rate; use only fiscal policy.

Total number of time periods = 40.

ATTACHMENT C

The background data for BARGIN is generated according to the following system. The two numbers input by the negotiators are A and B respectively. Then:

DEMAND FOR LABOR FUNCTION

$D = 100D - (10D)\,W$
Where $D = A + B$; W = the wage rate

PERCENTAGE CHANGE IN PROFITS

$P = (D/2) - 3$
Where $D = A + B$

PERCENTAGE INCREASE IN LABOR PRODUCTIVITY

$P = .1\,(AB)$

CURRENT LEVEL OF EMPLOYMENT

$E = 50\,(AB)$

PERCENTAGE INCREASE IN CPI = A

AVERAGE PERCENTAGE INCREASE RECENTLY OBTAINED BY OTHER UNIONS = B

CURRENT UNEMPLOYMENT RATE = A-.3, or B-.2, whichever is smaller.

THE DYNSIM SYSTEM
A SYSTEM FOR STUDENT INVESTIGATION OF ECONOMIC MODELS[1]

Paul N. Rappoport
Department of Economics
Ohio State University

INTRODUCTION:

The recent "computer boom" has served as a catalyst for the accelerated development of applied economics. Noteworthy has been the improvement in econometric methods. Equally significant has been the impetus given to the creation and simulation of economic models. The macroeconomist, for example, works with models of the economy to study the effects of alternative policy decisions. The microeconomist uses computer simulation to investigate specific market structures and forms. Assuming, therefore, that computer model construction and simulation have been established as an integral part of the discipline, we must now turn the focus of our attention towards the use of these tools in the teaching of the discipline.

This paper presents a project designed for integrating the advantages of the computer into the economics curriculum. Specifically, I shall describe and illustrate the system developed in the autumn of 1968 at Oberlin College.

Consultations between the Oberlin economists and computer systems analysts concluded that "textbook" economics could best be supplemented and enhanced by adopting a modeling-simulation approach. Like the macroeconomist in the example above, the student would thus be able to reduce complex economic interrelationships into a succinct hypothesis or model. He would be able to investigate the "properties" of his hypothesis and compare his analysis with that which was introduced in his textbook. He could develop the model of an economy and prescribe policy for the potential contingencies. He would have at his disposal a tool which could allow him to explore economic theory beyond the limitations of two-dimensional, graphical analysis and the simplifying constraints of first and second order difference equations. This approach offers concrete, analytical experience to supplement the often abstract or intangible theory.

DYNSIM (for dynamic simulation) offers the student two routes for his investigations. First, he can "invent" his own models. For example, a student might wish to study a macro-system in which movements in the money supply were adaptive--where the money supply was controlled by a unique decision rule. The inquisitive student might alter the "set" behavioral relationships between consumption and income. Whatever model the student chooses to study, he has complete control over the structure, variables, and running of the model. During the execution of the model, the student may alter parameter values or respecify any of the "equations" of the model. The flexibility and generality of the system reinforce the "desires" to alter "basic" assumptions, to derive unusual implications, and to "extend" an established model or theory.

Secondly, the student may investigate a model which had been previously constructed and stored. The example in Appendix A illustrates this approach. That specific model was constructed for an introductory course in macroeconomics at Oberlin. This route may be employed as a short-cut, and is most advantageous for large classes with relatively little sophistication in theory-building. Part of a homework assignment might be the analysis and comparison of three alternative lag structures of a previously stored model. Another assignment might inquire about the effects on the equilibrium level of income due to an increase in government expenditures. Corporate models including production control, sales forecasting, and inventory policy could be evaluated. The possibilities are unlimited.

It should be emphasized here that DYNSIM is quite easy to use. No previous mathematical or computer experience is required. The "rules" which a model must follow are "conversational", with little emphasis on formats or syntax. In order to present examples of this system, a brief overview of the system itself is first required. For the sake of exposition, detailed description is omitted. The primary points are illustrated by examples.

THE MECHANICS OF DYNSIM PROGRAMMING:

Three procedures define a DYNSIM problem: the specification of the model, the initialization of the variables and parameters of the model, and the creation of the controls which direct the simulation. A model consists of one or more equations, each which must conform to a few compositional rules. The evaluation of an equation occurs on the "right hand side". As an example of specifying a model, consider the problem of determining the level of income Y. Letting C denote consumption and I investment, one "possible" model to investigate would be:

1. $Y_t = C_t + I_t$
2. $C_t = aY_{t-1}$
3. $I_t = b(C_t - C_{t-1})$

The DYNSIM 'translation" of the above model is:

```
EQN 1 C(T)  = a * Y(T-1)
EQN 2 I(T)  = b * ( C(T) - C(T-1) )
EQN 3 Y(T)  = C(T) + I(T)
```

A variation of some interest of the previous model is:

1. $Y_t = C_t + I_t$
2. $C_t = aY_{t-1} + r^tY_{t-2}$
3. $I_t = b(C_t - C_{t-1})$

The DYNSIM representation is:

```
EQN 1 C(T)  = a * Y(T-1) + ( r ** T) * Y(T-2)
EQN 2 I(T)  = b * ( C(T) - C(T-1) )
EQN 3 Y(T)  =  C(T) + I(T)
```

These examples illustrate how "close" the DYNSIM representation is to the actual mathematical formulation. Currently, DYNSIM will accept a model consisting of up to 50 equations and 100 variables. If an equation is misspecified, DYNSIM acknowledges the mistake, indicates what is wrong and (in most cases) tells the user how the mistake can be corrected.

The second procedure is the data initialization section. Both parameters and variables must be initialized before each simulation. Consider the following one equation model:

```
EQN 1 Y(T) = a * Y(T-1) + b * ( Y(T-1) - Y(T-3) ) + G
```

The variables and parameters to be initialized are Y(T-1), Y(T-2), Y(T-3), a, b, and G. A possible specification for the data of the above model might be:

```
Y(T-1)=100.0,Y(T-2)=10.5,Y(T-3)=5.0,a=.5,b=.3333,G=1.0
```

The user must follow three insignificant rules. First, the period is necessary. Secondly, the comma after the value is required. Finally, imbedded blanks are not permitted.

```
EQN 1 Y(T)  = a * Y(T-1) + b * Y(T-2)

NO MORE MODEL CARDS

Y(T-1)=1.0,Y(T-2)=1.0,a=.5,b=1.5

NO MORE DATA CARDS
```

The final procedure is the model manipulation and control section. In this procedure, the user may: determine the length of the simulation, control when and how the data and/or structure of the model is changed, specify when and what type of output will be printed. There are three primary and two auxiliary commands available. The primary commands are:

1. TITLE allows the user the option of changing the title that appears with his output
2. GRAPH defines which variables are to be plotted against other variables
3. AFTER four AFTER commands are available:
   1. AFTER KK PERIODS, PRINT OUTPUT
   2. AFTER KK PERIODS, PRODUCE GRAPHS
   3. AFTER KK PERIODS, READ MODEL CARDS
   4. AFTER KK PERIODS, READ DATA CARDS

The use of the AFTER command in four situations is illustrated.

1. print initial data values
   AFTER 0 PERIODS, PRINT OUTPUT
2. run a simulation for 30 periods and print output
   AFTER 30 PERIODS, PRINT OUTPUT
3. run a simulation for 30 period,print output, and produce graphs
   AFTER 30 PERIODS, PRINT OUTPUT
   AFTER 0 PERIODS, PRODUCE GRAPHS
4. run a simulation for 10 periods and change a parameter value
   AFTER 10 PERIODS, READ DATA CARDS

DYNSIM follows a specific sequence; model specification, data initialization, and finally the simulation. Thus, if the user wanted to change the structure of a model, the following sequence would be necessary:

```
AFTER 30 PERIODS, PRINT OUTPUT

AFTER 0 PERIODS, READ MODEL CARDS

EQN 3.......

NO MORE MODEL CARDS

NO MORE DATA CARDS      (necessary even if no changes made)

AFTER.........
```

The final two commands provide for the saving and reading of models from computer storage. The commands are SAVE and READ.

SAVE (READ)

This option provides the student or instructor with a method to save (read) either a model or data or both on (from) a computer file. Following are two examples. The first will SAVE the model. The second will recall the model.

```
1 SAVE MODEL CARDS
  EQN 1 C(T) = a * Y(T-1) + b * Y(T-2)
  EQN 2 I(T) = r * ( C(T) - C(T-1) )
  EQN 3 Y(T) = C(T) + I(T)
  NO MORE MODEL CARDS
  SAVE DATA CARDS
  a=.25,b=.15,r=1.0,Y(T-1)=100.0,Y(T-2)=-50.0,C(T-1)=1.0
  NO MORE DATA CARDS
  ( Job control end of file card )


2 READ MODEL CARDS
  READ DATA CARDS
  AFTER 0 PERIODS, READ DATA CARDS
  Y(T-2)=50.0,a=.50
  NO MORE DATA CARDS
  AFTER .........
```

<u>SAMPLE APPLICATIONS OF DYNSIM</u>:

(A) The Samuelson model on the interaction between the multiplier and accelerator provides the starting point for the following example of a DYNSIM application. Suppose that an assignment for a class in macroeconomics is: "Study and comment upon the stability of the Samuelson model."

The student would be expected to examine the effects of changes in the two parameters, i.e. the MPC and the accelerator coefficient. A thorough analysis would, in addition, involve varying the form of the consumption function. Letting Y(T) be income at time T, C(T) be consumption at time T, G be exogenous government expenditures, <u>a</u> the marginal propensity to consume (MPC), and <u>b</u> the accelerator coefficient, his model is:

1. $Y(Y) = C(T) + I(T) + G$
2. $C(T) = aY(T-1)$
3. $I(T) = \underline{\bar{b}}(C(T) - C(T-1)$

The corresponding DYNSIM representation is:

card #

```
      1     Samuelson multiplier-accelerator model a = .5, b = 1
      2     EQN 1 C(T) = a * Y(T-1)
      3     EQN 2 I(T) = b * ( C(T) - C(T-1) )
      4     EQN 3 G(T) = 1
      5     EQN 4 Y(T) = C(T) + I(T) + G(T)
      6     NO MORE MODEL CARDS
      7     a=.5, b=1.0, C(T-1)= 10.0,Y(T-1) = 100.,G=1.0
      8     NO MORE DATA CARDS
      9     GRAPH Y(T) on TIME
      10    GRAPH C(T) on TIME
      11    AFTER 30 PERIODS, PRINT OUTPUT
      12    AFTER  0 PERIODS, PRODUCE GRAPHS
      13    AFTER  0 PERIODS, READ DATA CARDS
```

```
14   a = .75,b=1.0, Y(T-1)=100.0,C(T-1)= 10.0
15   NO MORE DATA CARDS
16   TITLE a changed from .5 to .75
17   GRAPH Y(T) on TIME
18   AFTER 30 PERIODS, PRINT OUTPUT
19   AFTER O PERIODS, PRODUCE GRAPHS
20   AFTER O PERIODS, READ MODEL CARDS
21   EQN 2 I(T) = b * ( C(T-1) - C(T-2) )
22   NO MORE MODEL CARDS
23   a = .5, Y(T-1)=100.0,C(T-1)=10.0,C(T-2)=5.0
24   NO MORE DATA CARDS
25   TITLE investment equation changed
26   GRAPH I(T) on TIME
27   GRAPH Y(T) on TIME
28   GRAPH Y(T) on I(T)
30   AFTER etc ........
```

The reader may have noticed certain inconsistencies in the syntax of the above statements. They are deliberate--intended to demonstrate the free format capabilities of DYNSIM.

(B) As another example, consider an economy described by the equations:

Consumption function $C_t = cY_{t-1}$
Investment function $I_t = i(Y_{t-1} - Y_{t-2}) + kR_{t-1}$
Commodity market equilibrium
$Y_t = C_t + I_t$

Demand for money $L_t = n + hY_t + fR_t$
Supply of money $M_t = m + vY_{t-1}$
Money market equilibrium $M_t = L_t$

The type of economy represented by these equations appears constantly in macroeconomic analysis. With a minimal amount of manipulation, the time paths of Y(T) and R(T), as determined by the dynamic IS and LM curves, could be shown and plotted.

There are many more possible systems which can be studied via DYNSIM. For instance, a variety of income expenditure models can be examined. Others include:

impact and delayed multipliers
inventory cycles
normative models (theory of economic policy)
models of aggregate demand (fiscal, monetary, labor markets)
full employment models
linear and non-linear growth models
production functions (Cobb - Douglas, etc.)
empirical models (Brookings, OBE models)
stability analysis
simulation of models of the firm
decision rule models

CONCLUSION:

Our experiences with DYNSIM support the contention that computers may be a tremendous asset in the teaching of economics. In no way, of course, can the computer be expected to replace the instructor. But rather, its utility can be be mobilized in support of the more traditional materials.

DYNSIM provides Oberlin students with the opportunity for valuable training in economic analysis, and offers them direct experience with relevant research methods. The students exercised great initiative in this pilot project at Oberlin. The models they created and explored represented a "continuing experiment" in applied economic theory.

DYNSIM was originally written for a batch-processing system because terminal set-ups were then not available at Oberlin. However, written in FORTRAN IV, DYNSIM was designed to be easily "portable". Only minor modifications are necessary to "transplant" DYNSIM to other environments (including terminal-based systems).[2]

APPENDIX A

Definition of variables in real terms

Y_t	GNP
DV	INVENTORY INVESTMENT
V_t	END OF PERIOD INVENTORIES
V_{t-1}	START OF PERIOD INVENTORIES
FS_t	FINAL SALES
T_t	GOVERNMENT TAXES
RED_t	GROSS BUSINESS SAVINGS
YD_t	HOUSEHOLD DISPOSABLE INCOME
CN_t	CONSUMPTION OF NONDURABLE GOODS AND SERVICES
KD_t	ACTUAL STOCK OF CONSUMER DURABLES, END OF PERIOD
C_t	TOTAL CONSUMPTION EXPENDITURES
I_t	GROSS BUSINESS FIXED INVESTMENT
K_t	ACTUAL STOCK OF BUSINESS CAPITAL, END OF PERIOD
G	GOVERNMENT PURCHASES OF GOODS AND SERVICES
SP_t	PERSONAL SAVINGS
KS_t	STOCK OF PERSONAL SAVINGS

FOOTNOTES

1. My colleague in this endeavor, Michael Rychener, deserves generous acknowledgment for his role in its success.

2. Decks are available from the author.

W2B/ECONOMICS

RAPPOPORT

OBERLIN COLLEGE
COMPUTER CENTER

```
// SEQNUM MMMMBB
//ECMOD JOB B52309MR,T=5,P=50,C=500,RYCHRAPP MP
//SYSPHS ACCESS ECONPHS2
// EXEC MASMODEL

                    EQN 1  FS(T) = FS(T-1) + A1 * ( FS(T-1) - FS(T-2) )

                    EQN 2    V(T) = A2 + A3 * FS(T)

                     EQN 3  Y(T) = FS(T) + A4 * ( V(T) - V(T-1) )

                     EQN 4 T    = A5 + A6 * Y(T)

                     EQN 5 RED(T) = A7 + A8 * Y(T)

                     EQN       6   YD(T) = Y(T) - T    - RED(T)

                     EQN 7  CN(T) = A9 + A10 * YD(T) + A11 * YD(T-1) + A12 * CN(T-1)

                     EQN 8        KD(T) = A13 * ( A14 * YD(T) + A15 * YD(T-1) + A16 * KS(T-1) )

                     EQN  9 CD    = A17 * ( KD(T) - KD(T-1) ) + A18 * KD(T-1)

                    EQN 10      KD(T) = KD(T-1) + CD    - A18 * KD(T-1)

                     EQN 11  C    = CN(T) + CD

                    EQN  12  SP    = YD(T) - C

                     EQN 13  KS(T) = KS(T-1) + SP

                      EQN 14  K(T) = A19 * ( A20 * Y(T) + A21 * Y(T-1) + A22 * Y(T-2) ) + A23 *   X

                      ( A24 * RED(T) + A25 * RED(T-1) + A26 * RED(T-2) ) + A27

                    EQN 15  I    = A28 * ( K(T) - K(T-1) ) + A29 * K(T-1) + A30

                    EQN 16 K(T) = K(T-1) + I    - A29 * K(T-1)

                    EQN  17 G    = A31

                      EQN 18  FS(T) = C    + I    + G

                    EQN 19 DV    = Y(T) - FS(T)

                     EQN 20 V(T) = V(T-1)  + DV

                     EQN 21  SG    = T    - G

                      EQN 22 VAR1 = I     + DV

                    EQN 23 VAR2 = SP    + SG    + RED(T)

                        NO MORE MODEL CARDS

                      FS(T-1)=800. , FS(T-2) = 800.,V(T-1)=400.,YD(T-1)=600.,CN(T-1)=500.

                    KS(T-1)=1000.,KD(T-1)=300.,Y(T-1)=800.,Y(T-2)=800.,RED(T-1)=100.,RED(T-2)=100.
```

100

100/a

dartmouth conference proceedings

100

OBERLIN COLLEGE
COMPUTER CENTER

```
                K(T-1)=2000..

                A1=0..A2=0..A3=.5,A4=0..A5=-20..A6=0.15,A7=20..A8=.1,A9=100..A10=.6667

                 A11=0..A12=0..A13=0..A14=0..A15=0..A16=0..A17=0..A18=0..A19=0..A20=0..

                 A21=0..A21=0..A22=0..A23=0..A24=0..A25=0..A26=0..A27=2000..A28=0..A29=.1.

                 A30=0..A31=100.

                NO MORE DATA VALUES

                PRINO INITIAL DATA VALUES

  FS(T)           FS(T-1)         A1              FS(T-2)         V(T)            A2              A3              Y(T)
0.0             0.80000000E 03  0.0             0.80000000E 03  0.0             0.0             0.50000000E 00  0.0

  A4              V(T-1)          T               A5              A6              RED(T)          A7              A8
0.0             0.40000000E 03  0.0            -0.20000000E 02  0.14999992E 00  0.0             0.20000000E 02  0.99999964E-01

  YD(T)           CN(T)           A9              A10             A11             YD(T-1)         A12             CN(T-1)
0.0             0.0             0.10000000E 03  0.66669995E 00  0.0             0.60000000E 03  0.0             0.50000000E 03

  KD(T)           A13             A14             A15             A16             KS(T-1)         CD              A17
0.0             0.0             0.0             0.0             0.0             0.10000000E 04  0-0             0.0

  KD(T-1)         A18             C               SP              KS(T)           K(T)            A19             A20
0.30000000E 03  0.0             0.0             0.0             0.0             0.0             0.0             0.0

  A21             Y(T-1)          A22             Y(T-2)          A23             A24             A25             RED(T-1)
0.0             0.80000000E 03  0.0             0.80000000E 03  0.0             0.0             0.0             0.10000000E 03

  A26             RED(T-2)        A27             I               A28             K(T-1)          A29             A30
0.0             0.10000000E 03  0.20000000E 04  0.0             0.0             0.20000000E 04  0.99999964E-0)  0.0

  G               A31             DV              SG              VAR1            VAR2            00000000        00000000
0.0             0.10000000E 03  0.0             0.0             0.C             0.0             0.0             0.0

                    AFTER 1 PERIOD,READ DATA

                    A31=120.

                    NO MORE

                  AFTER 30 PERIODS,PRINT OUTPUT,6
```

OBERLIN COLLEGE
COMPUTER CENTER

```
                   SAMPLE TITLE CARD

PERIOD       FS(T)          V(T)          Y(T)           T           RED(T)         YD(T)          CN(T)

    1       800.00        400.00        800.00        100.00        100.00        600.00        500.02
    2       800.02        400.01        800.02        100.00        100.00        600.01        500.03
    3       820.03        410.01        820.03        103.00        102.00        615.02        510.03
    4       830.03        415.02        830.03        104.51        103.00        622.53        515.04
    5       835.04        417.52        835.04        105.26        103.50        626.28        517.54
    6       837.54        418.77        837.54        105.63        103.75        628.15        518.79
    7       838.79        419.40        838.79        105.82        103.88        629.09        519.42
    8       839.42        419.71        839.42        105.91        103.94        629.56        519.73
    9       839.73        419.86        839.73        105.96        103.97        629.80        519.89
   10       839.88        419.94        839.88        105.98        103.99        629.91        519.96
   11       839.96        419.98        839.96        105.99        104.00        629.97        520.00
   12       840.00        420.00        840.00        106.00        104.00        630.00        520.02
   13       840.02        420.01        840.02        106.00        104.00        630.02        520.03
   14       840.03        420.02        840.03        106.00        104.00        630.02        520.04
   15       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   16       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   17       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   18       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   19       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   20       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   21       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   22       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   23       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   24       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   25       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   26       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   27       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   28       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   29       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   30       840.04        420.02        840.04        106.01        104.00        630.03        520.04
   31       840.04        420.02        840.04        106.01        104.00        630.03        520.04
```

OBERLIN COLLEGE
COMPUTER CENTER

```
                    SAMPLE TITLE CARD

PERIOD       I            K(T)          G            FS(T)         DV           V(T)          SG


   1       200.00       2000.00       100.00        800.02        -0.02        399.98        -0.00
   2       200.00       2000.00       120.00        820.03       -20.01        379.97       -20.00
   3       200.00       2000.00       120.00        830.03       -10.01        369.97       -17.00
   4       200.00       2000.00       120.00        835.04        -5.00        364.96       -15.49
   5       200.00       2000.00       120.00        837.54        -2.50        362.46       -14.74
   6       200.00       2000.00       120.00        838.79        -1.25        361.21       -14.37
   7       200.00       2000.00       120.00        839.42        -0.63        360.58       -14.18
   8       200.00       2000.00       120.00        839.73        -0.31        360.27       -14.09
   9       200.00       2000.00       120.00        839.88        -0.16        360.12       -14.04
  10       200.00       2000.00       120.00        839.96        -0.08        360.04       -14.02
  11       200.00       2000.00       120.00        840.00        -0.04        360.00       -14.01
  12       200.00       2000.00       120.00        840.02        -0.02        359.98       -14.00
  13       200.00       2000.00       120.00        840.03        -0.01        359.97       -14.00
  14       200.00       2000.00       120.00        840.04        -0.00        359.96       -14.00
  15       200.00       2000.00       120.00        840.04        -0.00        359.96       -13.99
  16       200.00       2000.00       120.00        840.04        -0.00        359.96       -13.99
  17       200.00       2000.00       120.00        840.04        -0.00        359.96       -13.99
  18       200.00       2000.00       120.00        840.04        -0.00        359.96       -13.99
  19       200.00       2000.00       120.00        840.04        -0.00        359.96       -13.99
  20       200.00       2000.00       120.00        840.04        -0.00        359.96       -13.99
  21       200.00       2000.00       120.00        840.04         0.0         359.96       -13.99
  22       200.00       2000.00       120.00        840.04         0.0         359.96       -13.99
  23       200.00       2000.00       120.00        840.04         0.0         359.96       -13.99
  24       200.00       2000.00       120.00        840.04         0.0         359.96       -13.99
  25       200.00       2000.00       120.00        840.04         0.0         359.96       -13.99
  26       200.00       2000.00       120.00        840.04         0.0         359.96       -13.99
  27       200.00       2000.00       120.00        840.04         0.0         359.96       -13.99
  28       200.00       2000.00       120.00        840.04         0.0         359.96       -13.99
  29       200.00       2000.00       120.00        840.04         0.0         359.96       -13.99
  30       200.00       2000.00       120.00        840.04         0.0         359.96       -13.99
  31       200.00       2000.00       120.00        840.04         0.0         359.96       -13.99
```

```
                                                                                    OBERLIN COLLEGE
                                                                                   COMPUTER CENTER

                     SAMPLE TITLE CARD

PERIOD      KD(T)           CD              KD(T)           C               SP              KS(T)           K(T)

    1         0.0             0.0          300.00          500.02           99.98         1099.98         2000.00
    2         0.0             0.0          300.00          500.03           99.99         1199.97         2000.00
    3         0.0             0.0          300.00          510.03          104.99         1304.95         2000.00
    4         0.0             0.0          300.00          515.04          107.49         1412.44         2000.00
    5         0.0             0.0          300.00          517.54          108.74         1521.18         2000.00
    6         0.0             0.0          300.00          518.79          109.36         1630.54         2000.00
    7         0.0             0.0          300.00          519.42          109.68         1740.22         2000.00
    8         0.0             0.0          300.00          519.73          109.83         1850.05         2000.00
    9         0.0             0.0          300.00          519.89          109.91         1959.96         2000.00
   10         0.0             0.0          300.00          519.96          109.95         2069.91         2000.00
   11         0.0             0.0          300.00          520.00          109.97         2179.88         2000.00
   12         0.0             0.0          300.00          520.02          109.98         2289.86         2000.00
   13         0.0             0.0          300.00          520.03          109.98         2399.85         2000.00
   14         0.0             0.0          300.00          520.04          109.99         2509.84         2000.00
   15         0.0             0.0          300.00          520.04          109.99         2619.82         2000.00
   16         0.0             0.0          300.00          520.04          109.99         2729.81         2000.00
   17         0.0             0.0          300.00          520.04          109.99         2839.80         2000.00
   18         0.0             0.0          300.00          520.04          109.99         2949.79         2000.00
   19         0.0             0.0          300.00          520.04          109.99         3059.78         2000.00
   20         0.0             0.0          300.00          520.04          109.99         3169.77         2000.00
   21         0.0             0.0          300.00          520.04          109.99         3279.76         2000.00
   22         0.0             0.0          300.00          520.04          109.99         3389.75         2000.00
   23         0.0             0.0          300.00          520.04          109.99         3499.74         2000.00
   24         0.0             0.0          300.00          520.04          109.99         3609.73         2000.00
   25         0.0             0.0          300.00          520.04          109.99         3719.72         2000.00
   26         0.0             0.0          300.00          520.04          109.99         3829.71         2000.00
   27         0.0             0.0          300.00          520.04          109.99         3939.70         2000.00
   28         0.0             0.0          300.00          520.04          109.99         4049.69         2000.00
   29         0.0             0.0          300.00          520.04          109.99         4159.67         2000.00
   30         0.0             0.0          300.00          520.04          109.99         4269.66         2000.00
   31         0.0             0.0          300.00          520.04          109.99         4379.65         2000.00
```

```
                              SAMPLE TITLE CARD

 ERIOD        VAR1                VAR2
   1          199.98              199.98
   2          179.99              179.99
   3          189.99              189.99
   4          195.00              195.00
   5          197.50              197.50
   6          198.75              198.75
   7          199.37              199.37
   8          199.69              199.69
   9          199.84              199.84
  10          199.92              199.92
  11          199.96              199.96
  12          199.98              199.98
  13          199.99              199.99
  14          199.99              199.99
  15          200.00              200.00
  16          200.00              200.00
  17          200.00              200.00
  18          200.00              200.00
  19          200.00              200.00
  20          200.00              200.00
  21          200.00              200.00
  22          200.00              200.00
  23          200.00              200.00
  24          200.00              200.00
  25          200.00              200.00
  26          200.00              200.00
  27          200.00              200.00
  28          200.00              200.00
  29          200.00              200.00
  30          200.00              200.00
  31          200.00              200.00

  NORMAL TERMINATION.
/&
```

EVALUATION OF STUDENT LEARNING AND INFORMATION UTILIZATION IN A COMPUTER-SIMULATED ECONOMIC MODEL

Jo-Anne Sharac and Darlene Russ
University of Michigan
Ann Arbor, Michigan

Computer-mediated econometric models are increasingly used in undergraduate and graduate economics courses. Two major reasons for this increased use are to raise student interest in economic theories and models and to provide a more dynamic model which approaches the complexity of the real world (Holbrook, 1970). In our investigation we have ignored an evaluation of the former reason and have attempted to concentrate on the latter. Our interest focused on attempting to determine how students' use of a computer model affects their economic understanding. To our knowledge there is little such evaluation (Feldhusen, 1969; Holbrook, 1970). The types of evaluation selected for examination were: 1) what the students learned from using the model and 2) the extent to which they utilized the information provided by the variables in the model.

Students' learning and information utilization were to be affected by their previous exposure to economic models and the tasks which they were instructed to perform in the work session. Diagram 1 shows the design of this investigation.

Diagram 1 - Design of Investigation

A. Learning given	1) Previous exposure to dynamic models
	2) Type of task.
B. Information utilization given	1) Previous exposure to dynamic models
	2) Type of task

Previous exposure to economic models refers to the type of material covered in the individual sections of the class. The sections from which the students were selected studied either the traditional static models or a dynamic model in addition to the static models.

The second factor affecting learning and information utilization was the task during the work session. The computer model in the work session may be utilized in two fashions. The student can either attempt to optimize and stabilize a particular variable(s) or he can manipulate the model in any desired fashion, including extreme changes in variables.

Within this framework our study attempted to answer the following questions:

1) Do those students given training on dynamic economic models perform better on the experimental model than those just exposed to static models? Do they use more of the variables?

2) Do those instructed to optimize one variable of the model (optimizing group) perform differently on the post-tests of the model than those instructed to look for the important relationships of the model in any way they prefer?

3) Which variables of the experimental model were used in making decisions? Is there any relationship between the number used and the score on the post-tests?

4) Control--Do the groups perform at about the same level on the pre-test questions on economic theory?

METHOD

Subjects: The students involved in the study were undergraduates enrolled in the introductory economics course at the University of Michigan. The study was characterized as a two-hour work session, during which time they would be able to run a computer model of a macroeconomy. Eighty-eight of the approximately 750 students participated in the study for pay ($2.00/hr). Included in this sample were all the students in the computer section of the course. Prior to their session in this study the computer section students had been exposed to approximately two weeks of theory and discussion about dynamic models. During this time they were divided into groups of four or five and each group assigned particular problems to attempt to understand by using a computer model that was not the one employed in this study. The class model was a scaled down version of a quarterly forecasting model developed by the Research Seminar in Quantitative Economics, Dept. of Economics, University of Michigan. None of the assigned problems dealt with welfare, the variable in which our optimizing groups were interested. After working on the problems each group brought the results to the class sessions to discuss the findings with the other groups. None of the groups were assigned the same problems. During the experimental session, the students worked on the model either alone or in pairs.

Design: A 2x2 factorial design investigated the effects of task instructions (learning vs. optimizing) and prior experience with a computer model on behavior during the work session and on success of answering questions of a test based on the model. The variable of prior experience relates to the student's experience with computer models of a macroeconomy.

Materials: There were four types of materials--a pre-test, an instruction booklet, the computer model, and the post-tests. The half-hour pre-test was a series of essay questions designed by one of the instructors to examine the students' knowledge of previously studied basic concepts and relationships such as disposable income, fiscal and monetary policies, etc., that would be useful in running the model.

The instruction booklet included a one page description of the model, definitions of all the twenty-one variables in the model, a past history of the model including numeric values for all variables for quarters 16 through 20, and task instructions. The task instructions for the students in the optimizing conditions were as follows: "Your task is to discover the relationships between the important variables in the economy. In order to do so, you must maintain welfare at as high a level as possible for at least six periods. At the end of a forty-five minute work session you will take a test of your knowledge of the model."

The instructions for the learning conditions included no restrictions about having to maintain welfare. Instead, these students were instructed to manipulate the economy in any desired manner in order to discover the important relationships.

The macro-economic model employed in this investigation was developed by Dr. Robert Holbrook (1970). It consists of four exogenous variables (nominal government purchases, nominal money supply, intercept of nominal tax function, slope of nominal tax function) and seventeen endogenous variables. Although Holbrook interrelated the economies of the various students through international trade and capital flow, we did not follow the same procedure. Instead, the economies run by our students were independent of one another. We did, however, retain the stable relationship between unemployment and inflation (Phillips curve).

A 360/67 IBM computer runs the model with the students working at remote terminals. The operating procedure is to type the numeric values for the exogenous variables for the following quarter. The computer then reprints the values in order to allow the students to correct any errors they feel they might have made. When the students indicate that the values are correct, the solution program runs, and the new values are stored in a file reserved for that particular economy. These new values for all the twenty-one variables are printed out at the terminal. The students then indicate whether they wish to run another quarter. If so, they rerun through the same procedure.

The post-test was designed to take no more than 30 minutes to complete. Multiple choice and true-false test items included questions on the relationships between the variables in the model and on economic theory related to the model. Open-ended questions attempting to learn the students' reasons for selecting the particular variables and strategies were also asked.

Procedure: Prior to working with the model all students were given the pre-test of economic theory which they were instructed to answer with brief responses, taking no more than 20 minutes. Upon completion of the text they were supplied with the instruction booklet which they were allowed 10-15 minutes to read. Those students working on the model in pairs were permitted to confer together.

After reading the instructions, they were informed that they would begin to run the economy with the twenty-first quarter. All students received the same history and economy. Running the economy consisted of supplying numeric values for the four exogenous variables which students were allowed to change, seeing the effects of these changes on the remaining seventeen endogenous variables, and as a result of the observed changes, deciding what values to assign to the variables for the next quarter. For each run, students were to check the variables they considered in order to make their decision about the values to supply for the four variables they controlled directly; and, after determining those values, they were to predict whether the variables used in making their decision would increase, decrease, or remain unchanged as a result of their changes. After the quarter was run they were to indicate what actually did happen to these variables. This procedure was followed for every quarter run.

After working with the model for a minimum of six quarters, they were given the post-tests. To answer the questions they were allowed to use any of the output they generated while working with the model. Those students who worked on the model in pairs were instructed to answer all post-test quesions separately.

RESULTS-DISCUSSION

At the moment we are in the process of coding an analyzing the data. Therefore the present results include only preliminary findings and tentative conclusions.

Regarding the information utilization variable few of the students in either the learning or optimizing groups utilized all the information supplied by the seventeen endogenous variables. Instead the students tended to concentrate on six or seven of these variables throughout the session. The variables used most often by the students were: GNP, welfare, inflation, unemployment, and consumption. This concentration might have resulted from the relatively limited time during which the students could work with the model.

In comparing the learning and optimizing conditions it appears that the students in both conditions operated in similar fashions. There were two dimensions to this similarity. First, whereas the task of the optimizing condition was to concentrate on influencing welfare, the task of the learning group was not limited to such a narrow focus. However, the students in the latter condition appeared to limit themselves to a concentration on one or two variables, primarily GNP and welfare. Secondly, both groups attempted to maximize the selected variables even though the students in the learning groups were not instructed to employ such a strategy.

Although the study was not concerned with student interest per se, the authors noted much enthusiasm on the part of the participants. Many of the students wished to extend the session and/or requested to work with the model in a second session.

FOOTNOTES

*Final results and conclusions will be available by the time of the Conference.

ACKNOWLEDGMENTS

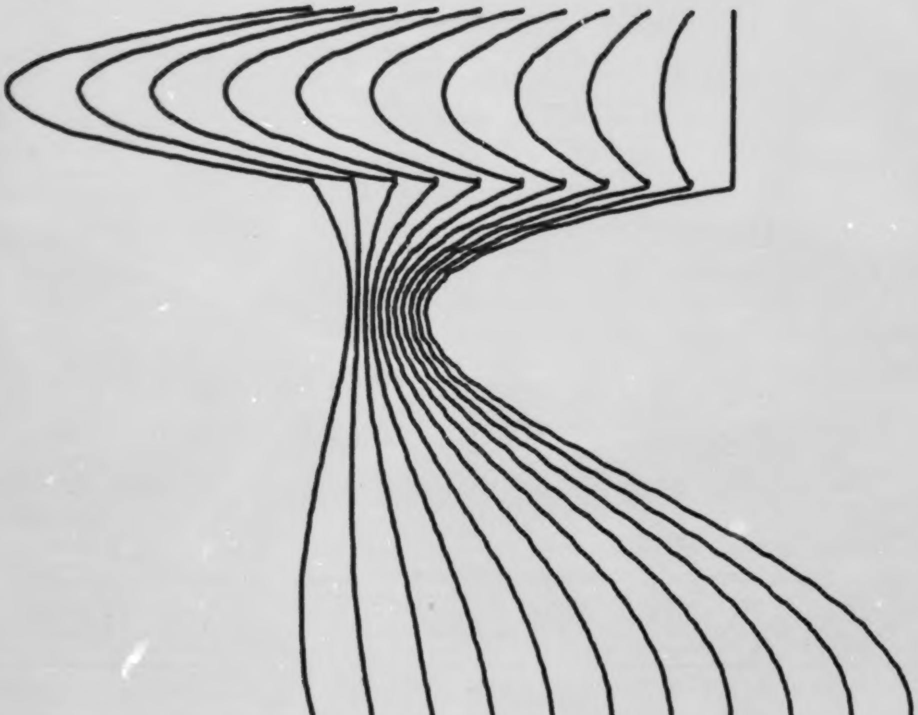
1. Preparation of this paper was partially supported by grants from the Air Force Office of Scientific Research, U.S. Air Force, AFOSR-68-1601, and from the Center for Research on Learning and Teaching, University of Michigan.

2. We are grateful to Dr. David Starks for his assistance in the preparation of this paper. Also, we thank Dr. Daniel Fusfeld for permitting us to conduct our study with his classes and Mr. Gordon Sellon for developing the economic tests.

BIBLIOGRAPHY

Feldhusen, J.E., A position paper on CAI research and development. A position paper for the "Computers in Education" project of the ERIC Clearinghouse on Educational Media and Technology, Stanford University, Stanford, California, June 1, 1969.

Holbrook, R.F., "The use of economic policy games as an aid to student understanding," Proceedings of a Conference on Computers in the Undergraduate Curricula. Iowa City: The University of Iowa, 1970.



John R. Merrill

1 : 1

TEACHING PROGRAMMING LANGUAGES AND OPTIMIZING NON-INTERACTIVE COMPUTER APPLICATIONS (NICA) IN THE CHEMISTRY CURRICULUM

Ronald W. Collins
Eastern Michigan University
Ypsilanti, Michigan 48197

In discussing the utilization of computers in chemistry education, it is appropriate to begin by presenting several schemes for the classification of each suggested application on the basis of both mode and type (1). According to mode all computer usage in the chemistry curriculum can be divided into either computer assisted instruction (CAI) or non-interactive computer applications (NICA). The CAI mode is interactive; i.e., it features a continuing student-computer dialogue in which the machine is serving as a surrogate teacher. An excellent example of CAI usage in chemistry is the PLATO system at the University of Illinois (2). The status of CAI in chemistry education has been reviewed recently by Tabbutt (3) and I will not attempt to discuss this area further. Instead, I will emphasize the alternate mode of computer utilization, namely NICA, which is more within the technological and economic reach of a greater number of academic institutions. In this mode students process data via the usual input-output cycle utilizing either batch card processing or remote teletype terminals. The computer does not instruct per se nor does it, in most instances, offer commentary on the validity of the results obtained; nevertheless, the pedagogical potential of a well-conceived and properly-implemented scheme of NICA is very high. A comparison of the characteristics of both CAI and NICA is given in Table I.

According to type of usage NICA can be divided into the following categories: data reduction, data simulation, data evaluation, data accumulation and optimization, and information retrieval. A brief summary of the features of each of these five types of computer usage is given in Table II. Since all five categories of usage are very important to the modern chemist, any up-to-date chemistry curriculum should at least expose its students to each of these computer capabilities. This could conceivably be accomplished via a single, specialized advanced course, but I believe that a broad integrated instructional plan involving systematic exposure across course lines is a better approach. The following factors, as presented in outline form, are, in my opinion, necessary for the development of an effective program for the educational use of computers in chemistry:

1. Stimulate student interest in computers, preferably during the freshman year.
    a. Provide a realistic awareness of computer capabilities and stress their relevancy to modern research and technology.
    b. Include programming and computer usage as a course requirement.

2. Provide the necessary communications techniques and skills for student interaction with the computer.
    a. Teach the use of teletype and/or keypunch machines for using "canned" programs.
    b. Teach programming language(s) via the course framework, non-credit optional lectures, or self-study tape recordings.

3. Develop an integrated program to expand and diversify the student's computer-related skills.
    a. Requires curriculum supervision and coordination to allow for continuation of computer usage in varying modes.
    b. Requires Department and University commitment to provide hardware and operating funds.
    c. Stress learning, not just relief from calculations. (lecture vs. laboratory).
    d. Requires access to numerous programs and/or prolific programmer(s).

4. Provide for capable assistance when needed.
    a. Faculty.
    b. Graduate Assistant for computer programming and applications.
    c. Computer Center organized for helping students.

5. Develop student awareness of both the cost and time economics of computer usage.
    a. Provide individual student budget allotments for a given term or course.
    b. Demand realistic use of the proper computing tool; i.e., computer vs. desk-top programmable calculator vs. electro-mechanical calculator.

Based on these guidelines, we have implemented a program for computer usage in chemistry at Eastern Michigan University which we believe has been relatively successful in meeting the stated requirements. Our approach provides for instruction in programming languages, extensive use of various types of NICA in a variety of courses, and adequate available help for students having computer related difficulties. The essential features of the EMU approach are given below in outline form:

1. Teach programming languages (FORTRAN, PIL, BASIC) and computer-related mechanical skills (keypunch machine, teletype), in the freshman year in general chemistry via:
   a. Using lecture time (at the discretion of the individual instructor).
   b. Non-credit lectures taught by the chemistry faculty (late afternoon and evening).
   c. Self-study tape recordings available in the library.

   A problem oriented approach to teaching programming languages is used; i.e., the rules of a given language are developed by analyzing available programs. For example, the simple FORTRAN programs shown in Figure 1 are used to teach the basics of formating, and the use of "DO" and "IF" statements. Following this basic instruction computer programming problems of the type shown in Figures 2 and 3 are assigned to the students.

2. Incorporate NICA into a wide variety of courses, both of the lecture-only type and those with scheduled laboratory sessions (see Table III for a summary of the courses involved).

3. Provide access to a departmental library of computer programs complete with a self-explanatory user description for each program and a keyword index to all available programs. The resources included in this library have been greatly expanded via the creation of the EMU Center for the Exchange of Chemistry Computer Programs (or EMU-CECCP), an inter-university exchange for educational computer programs.

4. Provide capable assistance in the form of the Graduate Assistant for Computer Programming and Applications, whose duties include:
   a. Consulting with the faculty on the development of clerical, instructional, or research programs.
   b. Delivering lectures on programming languages and computer-related mechanical skills.
   c. Conducting tours of the Instructional Computer Services facility.
   d. Consulting with both undergraduate and graduate students on programming problems.
   e. Serving as director of the EMU Library of Computer Programs.
   f. Coordinating the operation of the EMU-CECCP.

Somewhat unique and of particular pedagogical significance in the EMU scheme is the utilization of NICA in lecture-only courses. This answers what I feel has been a general failure of many departments in attempting to incorporate computer usage into their instructional program, namely, placing too much emphasis on the use of the computer solely for laboratory data reduction. This not only creates the image that the computer is just a large calculator, but it fails to utilize even the "pseudo-instructional" capabilities of the machine. Providing instruction in programming languages and incorporating programming projects into the course content is a very effective instructional technique in lecture courses ranging from first-year general chemistry (4) to senior level advanced inorganic (5). In particular, it can be cited that students gain a true mastery of the theory of any topic for which they write a program. Also, by requiring that the students write an accompanying description of each of their programs and make the programs available for use by their fellow students, these assignments serve to get the students more involved in their own instruction. In an advanced (senior-graduate level) course programming assignments serve the additional purposes of requiring an intensive analysis of the theoretical background of a given topic, encouraging the use of the modern research literature, and permitting the coverage of topics not actually discussed in lecture. For example, the programs shown in Figures 4 and 5 concerned with the calculation of lattice energies and the derivation of the formulas of boron hydrides, respectively, are both student-written programs on topics not discussed in depth in the Advanced Inorganic Chemistry lecture. Both programs involved considerable outside research by the student both to develop the theory and to locate appropriate data for execution. The lattice energy program (Figure 4) is also noteworthy in that it is "pseudo-interactive"; i.e., the student is asked questions regarding his choice of equations. The computer does not respond in an instructional manner, however, but merely reminds the user of what data will be required to carry out his instructions regarding the calculation.

In summary, we feel that our undergraduate students are obtaining a well-balanced exposure to the role of modern digital computers in chemistry and that they are prepared to utilize the computer fully whether they attend graduate school or work in industry. The specialized computer applications course has been particularly valuable to those students with strong interest in the role of computers in chemistry. Furthermore, this extensive involvement of computers in our curriculum serves a twofold purpose; i.e., it implements the actual teaching of the course content and it simultaneously teaches the student computer-related skills and techniques.

References

1. R. W. Collins, presented in part in a paper entitled "Teaching Programming Languages and Utilizing Digital Computers in the Chemistry Curriculum", 1970 Great Lakes Regional American Chemical Society Meeting, Fargo, North Dakota, June 1970.

2. S. G. Smith, "The Use of Computers in the Teaching of Organic Chemistry", J. Chem. Educ., 47, 608 (1970).

3. F. D. Tabbutt, "Computers in Chemical Education", Chem. and Eng. News, January 19, 1970.

4. R. W. Collins, presented in a paper entitled "Teaching Programming Languages and Utilizing Digital Computers in General Chemistry Lecture Courses", American Chemical Society Meeting, Los Angeles, California, April 1971.

5. R. W. Collins, presented in a paper entitled "Utilizing Computer Programming Projects in Advanced Inorganic Chemistry", American Chemical Society Meeting, Chicago, Illinois, September 1970.

6. R. W. Collins, presented in part in a paper entitled "A Versatile Computer Program for Processing Titration Data by the Derivative Method", American Chemical Society Meeting, New York, New York, September 1969.

7. R. W. Collins, presented in a paper entitled "The Use of Computers in Teaching X-Ray Crystallography to Undergraduate Chemistry Majors", 1970 Central Regional American Chemical Society Meeting, Columbus, Ohio, June 1970.

TABLE I

UTILIZATION OF COMPUTERS IN CHEMISTRY EDUCATION

Computer Assisted Instruction (CAI)	Non-Interactive Computer Applications (NICA)
1. Interactive with student-computer dialogue.	1. Non-interactive; normal input-output cycle.
2. On-line terminal communication.	2. Batch processing or teletype terminal communication.
3. Printed information and/or graphic display via CRT.	3. Printed information and/or plotted output.
4. Computer instructs via appropriate pedagogical logic cycle including remedial branching if appropriate.	4. Computer serves only to process input information rapidly without commentary on the accuracy or validity.
5. Emphasis on student mastery of concepts via tutorial drill at individualized pace.	5. Emphasis on applications and data reduction via rapid processing of input information.
6. All "canned" programs; student not involved in developing the program logic or writing the actual program.	6. Both "canned" and student-written programs can be used.

TABLE II

TYPES OF NICA IN CHEMISTRY

Category	Description
1. **DATA REDUCTION**	Mathematical processing of data, usually based on a laboratory experiment performed by the student.
2. **DATA SIMULATION**	Generation of tabular or graphical data based on theory rather than the student's own experiment; e.g., calculation and plotting of electron densities in atoms and molecules based on quantum mechanical equations.
3. **DATA EVALUATION**	Comparison of experimentally-obtained data to a library of reference data stored in the computer; e.g. identification of a compound via the comparison of its x-ray diffraction powder patterns to the complete ASTM library of reference powder patterns.
4. **DATA ACCUMULATION AND OPTIMIZATION**	Interfacing of a computer with an experimental apparatus; i.e., using the computer to control the actual performance of an experiment and to continuously collect and, if desired, monitor the quality of the data. Examples include on-line computer-automated x-ray diffraction equipment and mass spectrometers.
5. **INFORMATION RETRIEVAL**	Indexing and searching the modern research literature with the aid of computerized information retrieval techniques based on the use of descriptive keywords or chemical formulations.

TABLE III

NICA IN CHEMISTRY COURSES AT EASTERN MICHIGAN UNIVERSITY

Course	Type of Course	Processing	Canned or Student Written Programs	Examples of Types of Programs	Student Involvement
131 General Chemistry	lecture only	batch and terminal	both	gas law calculations; stoichiometry problems; electron configurations of atoms and ions	optional
180 General Chemistry Laboratory	lab only	batch and terminal	both	processing titration data[6] and kinetic data; least squares analysis of data	required
281 Quantitative Analysis	lecture/lab	batch and terminal	both	processing gravimetric and volumetric analysis data[6]; statistical treatment of results; simulation of titration data	required
432 Advanced Inorganic Chemistry	lecture only	batch	both	derivation of term symbols; calculation of lattice energies; boron hydride formulas; calculation of bond ionicity	optional
452 Biochemistry Laboratory	lab only	batch	canned	processing chromatography and spectral data	optional
462 Physical Chemistry Laboratory	lab only	batch and terminal	both	analysis of vibrational electronic spectra; x-ray diffraction powder data; gas kinetics	required
481 Instrumental Analysis	lecture/lab	batch	both	processing titration and radioactivity data; computer interfacing	required
490 Chemical Literature	lecture only	batch	canned	information retrieval and searching procedures	required
496 Special Topics (e.g., Magnetic Resonance)	lecture only	batch	canned	simulation and interpretation of NMR spectra	optional
530 X-ray Crystallography	lecture/lab	batch and terminal	both	reducing powder film data; indexing powder patterns; fourier analysis of single crystal intensity data; simulation of powder pattern data[7]	required
596 Computer Applications in Chemistry	lecture only	batch and terminal	both	primarily numerical methods of analysis	required

FIGURE 1

FORTRAN PROGRAMS USED FOR INSTRUCTION IN THE BASICS OF THE LANGUAGE

```
      DIMENSION NAME (10), NJOB (26), N DATE (4)
      READ (2,1) NN
    1 FORMAT (I2)
      DO 100 I=1, NN
      READ (2,11) NAME, NJOB,NDATE
   11 FORMAT ( 10A2,26A2,4A2)
      WRITE (5,12) NAME,NJOB,NDATE
   12 FORMAT (1H1//, 10A2, 26A2, 4A2//)
      READ (2,2) SN,R,T,P,PMAX,DELP
    2 FORMAT (6F10.5)
   15 V=(SN*R*T)/P
      WRITE(5,3) SN,T,P,V
    3 FORMAT (4F15.5)
      IF (P-PMAX) 10,100,100
   10 P=P+DELP
      GO TO 15
  100 CONTINUE
      CALL EXIT
      END
```

1. Simple Gas Law Program

```
      DIMENSION NAME(10), NJOB(26), NDATE(4)
      READ (2,1) NN
    1 FORMAT (I2)
      DO 100 I=1,NN
      READ (2,11) NAME, NJOB, NDATE
   11 FORMAT (10A2, 26A2, 4A2)
      WRITE (5,12) NAME, NJOB, NDATE
   12 FORMAT (1H1//, 10A2, 26A2, 4A2//)
      READ (2,2) DC, DCMAX, DCDEL
    2 FORMAT(3F 10.5)
   15 DF= (1.8*DC)+32
      DK = DC + 273.15
      WRITE (5,3) DC, DF, DK
    3 FORMAT (3F 15.5)
      IF (DC-DCMAX) 10, 100, 100
   10 DC = DC + DCDEL
      GO TO 15
  100 CONTINUE
      CALL EXIT
      END
```

2. Simple Temperature Conversion Program

FIGURE 2

SAMPLE COMPUTER PROGRAMMING ASSIGNMENTS FOR FRESHMAN STUDENTS IN GENERAL CHEMISTRY

I. Introductory Problems (for more difficult versions of the same problems see the back side of this page).

1. Using the specific heat data given below, write a computer program which will calculate individual values of Dulong and Petit's constant for each element listed. Print out in tabular form the chemical symbols of the elements, their atomic weights, the specific heats, and the calculated values of the constant.

element	specific heat (cal/g)	element	specific heat (cal/g)
Bi	0.0288	Te	0.0912
Pb	0.0293	Cu	0.0949
Au	0.0298	Ni	0.1035
Pt	0.0314	Fe	0.1100
Sn	0.0514	Co	0.1498
Ag	0.0557	S	0.1880
Zn	0.0927		

2. Assume that oxygen gas has been collected over water in a series of experiments run at 10°, 20°, 30°, 40°, 50°, 60°, 70°, 80°, and 90°C. In each of the separate experiments (one at 10°C, one at 20°C, etc) the volume of "wet gas" was exactly 100.00 ml and the pressure was 800 torr. Using the vapor pressure data for H_2O given below, write a computer program which will calculate the volume of dry oxygen gas at STP for each experiment. Print out in tabular form the V_1, P_1, P_{H_2O}, P_{O_2}, T_1, P_2, T_2 and V_2 for each experiment.

t(°C)	v.p. H_2O(torr)	t(°C)	v.p. H_2O(torr)
10°	9.1	60°	149.4
20°	17.5	70°	233.7
30°	31.8	80°	355.1
40°	55.3	90°	525.8
50°	92.5		

More Difficult Problems (expanded versions of the problems on the other side of the page.)

1a. Include in program 1 the average value of the Dulong and Petit constant based on the thirteen individual results. Using the proper formulas calculate also the standard deviation of the result.

2a. Include in program 2 the added capability of varying the initial pressure; i.e., do all nine dry gas volume calculations for the given temperatures at one initial pressure, then increment the pressure and repeat the nine calculations. Do this loop for approximately 10 different pressures spanning a range of about one atmosphere.

FIGURE 3

SAMPLE COMPUTER PROGRAMMING ASSIGNMENTS FOR FRESHMAN STUDENTS IN GENERAL CHEMISTRY

I. Introductory Problems (for more difficult versions of the same problems see the back side of this page).

1. Select a combustion reaction (e.g., $C_3H_8 + 5\ O_2 \rightarrow 4\ H_2O + 3\ CO_2$), and write a computer program which will calculate the mass and the volume (assuming gaseous products at STP) of both H_2O and CO_2 produced for different masses of the hydrocarbon reactant. Calculate also the weight of oxygen gas required to react with each weight of the hydrocarbon. Allow the mass of the hydrocarbon to vary over a range of approximately ten moles; e.g., from 0.1 mole to 10.0 moles in increments of 0.1 mole. Print the results in tabular form beneath the chemical equation, allowing enough space under H_2O and CO_2 to record both mass (g) and volume (ℓ or mℓ).

2. Write a computer program which will give the electron configuration in s, p, d, f notation for each of the first 36 elements on the periodic table. Restrict your input to the symbol of the element and the total number of electrons only; i.e., do not merely read in the configuration in "A" format and then write it again in the output. The program must deduce the proper configuration via a series of appropriate tests based on the Aufbau Principle. Print the results as a series of numerals in one row beneath headings of the form 1s, 2s, 2p, etc.

II. More difficult problems (expanded versions of the problems on the other side of the page).

1a. Generalize program 1 in order to handle a series of calculations based on different chemical equations. Use a general model equation of the form $aA + bB + cC \rightarrow dD + eE + fF$ and input the appropriate numerical values of the balancing coefficients a, b, c, etc., along with the chemical formulas of A, B, C, etc. Be sure to arrange the input so that the same general symbol, e.g., F, is always isolated for calculation even though it may be a reactant in a given equation. Execute one weight calculation for each of ten equations and print the results in tabular form under the general model equation as a heading.

1b. Expand program 1a to include the capability of deciding which is the limiting reagent when the weights of two reactants are specified. For example, how many grams of $CaCO_3$ are produced from the reaction of 56g. CaO with 50g. CO_2 according to the equation $CaO + CO_2 \rightarrow CaCO_3$? In this case CaO is the limiting reagent and must be used as the basis for the calculation; i.e., there is excess CO_2. Print the results as specified in 1a.

2a. Expand the program for predicting electron configurations to include the following: proper notation for the exceptions encountered at elements number 24 and 29; and, prediction of the electron configuration for at least one ion of each of the 36 elements; e.g., Na^+, Cl^-, etc. Print the results as specified in 1, but include the ions also.

FIGURE 4

OUTPUT FOR THE "PSEUDO-INTERACTIVE" LATTICE ENERGY PROGRAM

```
EQUATION FOR LATTICE ENERGY
DO YOU WANT

1= THE NAMES PRINTED OUT
2= THE FORMULAS PRINTED OUT
3= BOTH NAMES AND FORMULAS PRINTED OUT
?1
(1) BORN-LANDE EQUATION

(2) BORN-MAYER EQUATION

(3) REDUCED BORN-MAYER EQUATION

(4) EXTENDED BORN-MAYER EQUATION

(5) LADD AND LEE EQUATION

WHAT EQUATION DO YOU WANT TO RUN
?3

INFORMATION NEEDED FOR CALCULATION

A-AVOGADRO'S NUMBER (6.023 E+23)

M-MADLUNG CONSTANT

Z1,Z2-VALENCE OF IONS

E-CHARGE ON THE ELECTRON (4.8021 E-10)

RO-EQUILIBRIUM SEPARATION DISTANCE

P-CONSTANT

 DO YOU WANT LATTICE ENERGY CALCULATED
1=YES, 2=NO
?1

LATTICE ENERGY FOR
?MG(;)''+),F(-)

WHAT IS THE MADLUNG CONSTANT VALUE
?1.748

VALENCE OF IONS z1 & Z2
*Z1*Z2*
? 1  1

VALUE OF RO & P
*    RO    *    P    *
? 1.36        0.345
 LATTICE ENERGY FOR MG(+),F(-)

AS CALCULATED USING THE

(3)REDUCED BORN-MAYER EQUATION

LATTICE ENERGY= -   205.00000
```

FIGURE 5a

COMPUTER PROGRAM FOR THE PREDICTION OF THE STRUCTURAL FORMULAS OF BORON HYDRIDES

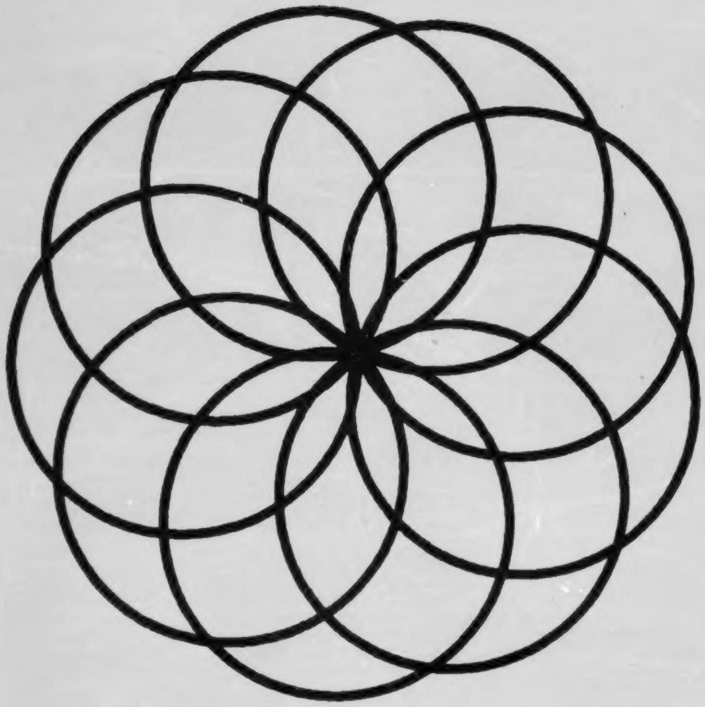
```
      DIMENSION X(25),S(25),T(25),Y(25)
C********** PREDICTION OF STRUCTURAL FORMULAS OF BORON HYDRIDES ****************
C********** REF.  BORON HYDRIDES  BY W. N. LIPSCOMB, 1963***********************
      I=0
      K=0
      WRITE(61,507)
  507 FORMAT(1H1,15X,*PREDICTION OF STRUCTURAL FORMULAS OF BORON HYDRIDE
     1S USING THE METHOD DEVELOPED BY LIPSCOMB*//)
      WRITE(61,500)
  500 FORMAT(25X,*POSSIBLE COMBINATIONS OF BONDS*//)
      READ(60,101)N
  101 FORMAT(I3)
  901 CONTINUE
C********** P IS THE NUMBER OF BORONS IN THE COMPOUND *************************
C********** F IS THE NUMBER OF HYDROGENS IN THE COMPOUND ***********************
C********** C IS THE NET CHARGE OF THE COMPOUND ********************************
      READ(60,100)P,F,C
  100  FORMAT(3F10.5)
      WRITE(61,509)P,F,C
  509 FORMAT (1H ,15X,*THE COMPOUND BEING STUDIED HERE IS  B*,F3.0,*H*
     1F3.0,* WITH A CHARGE OF *,F3.0//)
      DO 199 I=1,25
      T(I)=0
      Y(I)=0
      S(I)=0
  199 X(I)=0
      Q=F-P+C
      L=0
      A=P-Q/2-C
      D=ABS(A)
      K=ABS(A)+1
      IF(A)903,902,903
  903 CONTINUE
      DO 201 I=2,K
      Y(I)=L+1
  201 L=L+1
      DO 203 I=1,K
      T(I)=D
  203 D=D-1
  902 CONTINUE
      DO 204 I=1,K
  204 S(I)=ABS(P+C-T(I))
      DO 205 I=1,K
  205 X(I)=ABS(Q+C-S(I))
      WRITE(61,506)
  506 FORMAT(25X,*T*,9X,*Y*,9X,*S*,9X,*X*/)
      WRITE(61,501) (T(I),Y(I),S(I),X(I),I=1,K)
  501 FORMAT(20X,4F10.0)
      N=N-1
      IF(N)900,900,901
  900 CONTINUE
      WRITE(61,504)
  504 FORMAT (15X,*T IS THE NUMBER OF THREE CENTERED BORON BONDS*/)
```

FIGURE 5b

CONTINUATION OF THE BORON HYDRIDE PROGRAM

```
      WRITE(61,505)
  505 FORMAT (15X,*Y IS THE NUMBER OF BORON TO BORON BONDS*/)
      WRITE(61,503)
  503 FORMAT (15X,*X IS THE NUMBER OF BH TWO GROUPS*/)
      WRITE(61,502)
  502 FORMAT (15X,*S IS THE NUMBER OF HYDROGEN BRIDGE BONDS*//)
      WRITE(61,510)
  510 FORMAT(15X,*THE MOST SYMMETRICAL STRUCTURE, OBTAINED FROM THE LIST
     1OF POSSIBLE COMBINATIONS, IS THE CORRECT STRUCTURE*//)
      WRITE(61,508)
  508 FORMAT (15X*REFERENCE.  BORON HYDRIDES , W. N, LIPSCOMB, 1963*//)
      CALL EXIT
      END
```

```
               THE COMPOUND BEING STUDIED HERE IS  B  4H 10 WITH A CHARGE OF    0

                       T        Y         S         X

                          1        0         3         3
                          0        1         4         2
               THE COMPOUND BEING STUDIED HERE IS  B  3H  8 WITH A CHARGE OF   -1

                       T        Y         S         X

                          2        0         0         3
                          1        1         1         2
                          0        2         2         1
               THE COMPOUND BEING STUDIED HERE IS  B  2H  6 WITH A CHARGE OF    0

                       T        Y         S         X

                          0        0         2         2
               THE COMPOUND BEING STUDIED HERE IS  B  2H  7 WITH A CHARGE OF   -1

                       T        Y         S         X

                          1        0         0         3
                          0        1         1         2
               THE COMPOUND BEING STUDIED HERE IS  B 10H 15 WITH A CHARGE OF   -1

                       T        Y         S         X

                          9        0         0         3
                          8        1         1         2
                          7        2         2         1
                          6        3         3         0
                          5        4         4         1
                          4        5         5         2
                          3        6         6         3
                          2        7         7         4
                          1        8         8         5
                          0        9         9         6
```



James Cruce

DIRECT CLASSROOM USE OF A TIME-SHARING COMPUTER IN CHEMICAL EDUCATION

Philip E. Stevenson
Worcester Polytechnic Institute
Worcester, Massachusetts

I. General Considerations

There are many topics in the undergraduate chemistry curriculum where complicated mathematical formulas are necessary for quantitative descriptions of phenomena. These topics range from ionic equilibrium in freshman chemistry to advanced areas in quantum mechanics and statistical mechanics. While teaching these topics, the instructor equipped only with the conventional blackboard and chalk simply cannot demonstrate quantitative application of these formulas, except perhaps by reference to previously calculated quantities, or by the use of hand-waving approximations (as in ionic equilibria), or by the use of very simple idealized cases (such as the Ideal Gas Law).

However, with the use of special computer programs, previously written, debugged, and loaded into a time sharing computer system (such as the RCA Spectra 70/46 at WPI), an instructor with a remote time sharing terminal at his side can demonstrate these quantitative calculations to his classes. He need only type in a line or two of input and wait a moment for the results. The student question, "What happens if we change the value of ... to ...?" is easily answered.

In this paper I will discuss a number of such programs that I have written, some with the assistance of WPI undergraduates. Completed programs include "DIATH2", for computation of the binding energies of H_2 and H_2^+ , "SHMO" which performs Huckel MO calculations, "ACID" and "BRSE" for simulation of weak acid-strong base, and weak base-strong acid titrations, respectively, "GASLAW" for calculation of pressures by van der Waals' law, and "FEF" for calculation of free energies of reaction for arbitrary temperatures. Programs in development of contemplated include "SALT" for simulation of precipitation titrations and programs in the areas of reaction kinetics and statistical mechanics. First, however, I would like to discuss some of the general aspects of writing such programs.

To begin, one must choose a computer language. At WPI, this choice is among BASIC, FORTRAN IV, and TFOR (time-sharing FORTRAN). Each language has its own advantages and disadvantages. FORTRAN IV is computationally the most powerful of the three. This computation power was necessary for "DIATH2" which required double precision arithmetic for integral evaluation and for "SHMO" which utilizes "EIGEN" from the IBM Scientific Subroutine Package for matrix diagonalization. "ACID" and "BASE" are in FORTRAN IV because they were developed from a previous batch program written in WATFOR. In addition, source programs in FORTRAN IV are normally in card form and can be safely stored and easily be edited, duplicated and distributed. FORTRAN IV has three serious disadvantages, however. Any change in a program requires a batch run, so one must wait for whatever turnaround time one's computer center imposes, in order to check the results of the changes. Input format requirements are fairly rigid. For instance, one must right-justify all exponential and integer input. Finally FORTRAN IV load modules in the RCA system are large and non-sharable and sometimes generate paging problems. "DIATH2" for instance has a load module that is close to 100K bytes long. Four users executing "DIATH2" simultaneously would be utilizing about 40% of the available drum pages, and these pages would be swapping in and out of core often enough to decrease significantly the CPU efficiency. TFOR and BASIC on the other hand are not as computationally powerful as FORTRAN IV, but are interactive, so that programming errors can be immediately detected and remedied, have relatively few input restrictions, and use much less core and drum space for object coding.

Having chosen a language, one must next obtain a properly executing algorithm. Often one already has this in a batch program or a program obtained from elsewhere. Then one must devise an input-output scheme that is suitable for interactive execution of the program and suitable for the kind of terminal which one will be using. My personal preference is for teletype because it is the cheapest kind of terminal and because it produces hard copy. Its disadvantages are noisy operation and slow output (10 characters per second). In any event, one wishes to program for minimum input, and relatively small amounts of output. One should keep in mind that the interactive nature of such programs allows one to use output from an earlier part of the program to influence input to a later part of the program. If one has written his program in FORTRAN IV, one must provide some way of aligning input to the expected input format. I do this by outputting strings of characters just ahead of input lines which show me where to place the input. Finally, it is a good idea to have the program type out instructions for its own use.

The kind of problem one would choose for this sort of interactive program ideally would use a very small number of parameters as input, do lots of computation, and generate a modest number of significant results. I feel that the specific programs which I will now discuss more or less meet these criteria.

II. "DIATH2"--A Program for Demonstrating Chemical Bonding in Hydrogen

H_2 and H_2^+ are traditionally the first molecules encountered by the student of quantum chemistry. At some point in his introduction to the subject, he sees his instructor dutifully derive formulas for the energies of the valence bond function for H_2 (1) and the molecular orbital functions of H_2^+(2) and H_2 (3). He may even encounter the Weinbaum (4) and configuration interaction treatments. However, when it comes to numerical results, all he encounters are tables and perhaps a dissociation curve or two. He never sees nor participates in the process of calculating these results. If for this lack of participation he concludes that quantum mechanics is a "black art", he is not without some justification.

However, with the aid of a computer and a program, "DIATH2", described here, it is possible for the quantum chemistry teacher to add numerical results to his classroom discussion of H_2 and H_2^+ . Using a remote terminal connected to a time-sharing computer, he may enter arbitrary values of the internuclear distance, R, and 1s orbital exponent, ζ, and obtain the resulting energies within seconds. He could enter the "book" values of R and ζ, or with somewhat more expenditure of time, search for their optimum values. He could assign the construction of optimized dissociation curves as homework.

The program treats the various wavefunctions for H_2 and H_2^+ which can be constructed from 1s orbitals centered on each nucleus, namely the valence bond function of H_2, the molecular orbital functions for H_2^+ and H_2 and the simplest configuration interaction function for H_2 . The Weinbaum treatment is omitted, since it is formally identical to the configuration interaction treatment. The program, written entirely in FORTRAN IV, takes the input values of R and ζ, converting R into atomic units, if necessary, and evaluates all the necessary integrals involving components of the Hamiltonian operators over 1s orbitals (Equation 1) on the two nuclei.

$$1s = \zeta^{3/2}\pi^{-1/2}e^{-\zeta\rho} \qquad (1)$$

The program then evaluates the electronic, total, and binding energies for the valence bond wave function, the molecular orbital wave functions for H_2^+ and H_2, and the configuration interaction wave function[2]. At the user's option, any or all of the output may be typed at the terminal for immediate inspection. Any output not typed is printed off line for later use. An auxiliary subroutine "PARAB", is available which constructs a parabolic curve when given three points. It also gives the extreme point of the curve. This is useful when one is optimizing for ζ or R and desires a quadratic interpolation. The use of the input distance R = 0.0 results in output of the electronic energies of He^+ and He.

I have used this program in my Molecular Orbital Theory course. Student enthusiasm was high, and number of points were brought out and clarified which are often confusing or not appreciated. These points included the incorrect dissociation of the molecular orbital function, the united atom picture for electronic energy, and the role of orbital contraction. Application of the variation theorem was rather effectively demonstrated. However, in the course of an eighty minute class period, there was only time enough for a thorough discussion of H_2^+. Thus, in future years, I plan to assign homework problems involving student use of the program, as well as use it for demonstration purposes.

III. Ionic Equilibria Programs

Ionic equilibrium is traditionally one of the most difficult topics in the freshman chemistry syllabus. The combination of complex relationships among the variables, the need for such complicated mathematical techniques as the method of successive approximations, and the need to recognize which variables can be neglected in which equations proves to be the downfall of many a student. Demonstration calculations, though pedagogically necessary, are tedious and time consuming for an instructor to perform in class.

We are writing a series of demonstration programs which simulate titrations. "ACID" computes a pH vs. volume of base added curve for the titration of a monobasic, dibasic, or tribasic weak acid with a strong base. "BASE" is the analogous program for a monoacidic, diacidic, or triacidic weak base. Both "ACID" and "BASE" can also give concentrations of all substances at any point in the titration. "SALT", which is still in development, will simulate precipitation titrations in which the kinds of ions produce slightly soluble 1:1, 2:1, 3:1, or 3:2 precipitates.

One might better appreciate the value of a simulated titration if one recalls that titrations unify the problems of weak acid and base pH's, buffers, hydrolysis, salt solubilities, and common ion effects. A detailed discussion of "ACID" will help clarify this. One deals with the equilibrium of the weak acid HA with its ions;

$$HA(aq) = H^+ + A^- \qquad (2)$$

$$\frac{[H^+][A^-]}{[HA]} = K_a \qquad (3)$$

the self dissociation of water;

$$H_2O = H^+ + OH^- \qquad (4)$$

$$[H^+][OH^-] = K_w = 1.0 \times 10^{-14} \qquad (5)$$

the conservation of the initial amount of acid;

$$[HA] + [A^-] = F_a\left(\frac{V_a}{V_a+V_b}\right) \qquad (6)$$

and the conservation of electric charge;

$$[H^+] + [Na^+] = [OH^-] + [A^-] \qquad (7)$$

The Na^+ ion concentration is known to be:

$$[Na^+] = F_b\left(\frac{V_b}{V_a+V_b}\right) \qquad (8)$$

Equations (6), (7), and (8) assume that a given point in the titration is attained by mixing V_a ml of weak acid of formality F_a with V_b ml of NaOH of formality F_b. The program, in fact, assumes $V_a = 50.0$ ml and accepts F_a, F_b and K_a as input parameters. V_b is the independent variable in the titration and is automatically incremented from 0.0 ml to a point 20% past the equivalence point. The algebraic solution of Equations (3), (5), (6), (7), and (8) is programmed. The variables, (OH^-) and (A^-) are thus eliminated from Equation (7) and the resulting cubic equation in (H^+) is solved by the Newton-Raphson method.

One would, of course, never go through this complete solution in a hand calculation, but would instead neglect certain quantities in Equations (6) and (7) and then plug the equations into Equation (3), resorting to the method of successive approximations as a last resort. However, most freshmen lack the chemical intuition to be able to know what quantities to neglect and when to neglect them. Since "ACID" calculates the concentrations of all the ions in solution without neglecting any of them, a freshman chemistry instructor can use it in class to give his students some feeling for the numbers resulting from ionic equilibrium calculations. Thus students could begin to acquire the chemical intuition which they need in order to master the subject of ionic equilibrium.

I have used "ACID" experimentally in my two recitation sections in CH102 (freshman chemistry) at WPI. An ASR-33 teletype was set up at the side of a 30 seat classroom using a TV camera and monitor to show the results to the class. Communication with the RCA Spectra 70/46 computer at the Worcester Area College Computation Center was by telephone. I spent approximately 10 minutes at the beginning of class discussing Equations (2) through (8). Then I obtained the titration curve of 50.0 ml of 0.1F acetic acid with 0.1F NaOH and drew attention to the equivalence point pH of 8.72. The specific problems of the weak acid solution, the 1:1 buffer and the hydrolysis of the salt of a weak acid problem were then demonstrated by requesting computation of the concentrations of all species in solution at 0.0 ml, 25.0 ml, and 50.0 ml respectively in the titration. For these problems, all the standard approximations are valid and the program convinced the students that the concentrations of neglected species are indeed at least several orders of magnitude smaller than those of the major species. There remained time for only one additional titration. In one class, titration of a moderately strong weak acid ($K_a = 5 \times 10^{-3}$) was demonstrated; in the other the titration of a very very weak acid ($K_a = 1.0 \times 10^{-12}$) was demonstrated. In these two cases at least one approximation breaks down in each of the standard problems. For example, in the hydrolysis of the salt of the very very weak acid, the self-ionization of water (Equations 3 and 4) is significant. These breakdowns were clearly demonstrated.

Student response to the program is difficult to gauge at the time of this writing (the third day following the demonstration). The students asked very few questions and there was little spontaneous response in class. After class, however, one student asked me how to execute the program from the student teletypes; another complained of poor legibility of the numbers on the screen (a minor change in format will alleviate that problem); and a third said the demonstration was fascinating. The real test will be whether the demonstration enables the students to show an improved performance on their examinations. My feeling is that this will happen.

IV. Other Demonstration Programs

a. Quantum Chemistry

I have written an interactive program, "SHMO", which performs simple Hückel Molecular Orbital calculations. Input consists of the matrix dimension, number of π electrons, and the Hamiltonian matrix elements in units of β. Teletype output consists of eigenvalues, charge densities, and bond orders for bonded atom pairs. In addition, a complete output list is generated off line. The program makes use of the SSP matrix diagonalization subroutine "EIGEN". In addition, my research level program, "NEMO" (for Non Empirical Molecular Orbitals (5)) is set up for interactive teletype execution. Here too, a selection of output is typed on teletype, and the complete output list is generated off line. I make use of these two programs and "DIATH2" in my quantum theory course.

b. Statistical Mechanics

I have recently assigned a freshman honors lab student to program the following problem; Given a fixed number of identical indistinguishable particles possessing a fixed total energy, and free to move in a spectrum of evenly spaced one particle energy levels; compute the probability distribution of particles in these levels. This program is intended to generate a Boltzmann distribution by simple counting of particles and will compute distributions for large values of N and E following a hand calculated demonstration for small numbers such as $N = 3$ particles and $E = 6$ units of energy. The energy spectrum is assumed to consist of evenly spaced levels with separations of one unit starting at $\varepsilon = 0$.

c. Thermodynamics

One of my students has just completed a program, "GASLAW", which computes the Ideal and van der Waals' pressures of gases, given T, V, and N, as input. A table of parameters a and b for common substances is stored in the program, but the user can supply his own values if he wishes. This program can be used to provide a quantitative basis for discussions of deviations from the Ideal Gas Law.

She has also nearly completed a program, "FEF", which tests the validity of the approximation

$$\Delta G = \Delta H^\circ + T\Delta S^\circ \qquad (9)$$

where ΔG is the free energy of a chemical reaction at an arbitrary temperature T and ΔH° and ΔS° are the standard ($T = 298^\circ K$) changes in enthalpy and entropy respectively. The test is through comparison with the more accurate

$$\Delta G = \Delta H(T) + T\Delta S(T) \qquad (10)$$

where heat capacity data are used to evaluate $\Delta H(T)$ and $\Delta S(T)$ explicitly for the given temperature T. A table of heat capacity coefficients is stored in the program.

Another student is working on a program incorporating van der Waals' law and the criterion of inherent stability to generate P vs. V isotherms to demonstrate the phenomenon of condensation of a gas to the liquid state.

d. Chemical Kinetics

Another student is starting a project to simulate the time development of chemical reactions for a variety of reaction mechanisms. He has already written a program for the one-step, first-order, irreversible reaction

$$A \xrightarrow{k_a} B \tag{11}$$

and is working on the consecutive two-step reaction

$$A \xrightarrow{k_a} B \xrightarrow{k_b} C \tag{12}$$

where both reactions are first order and irreversible.

V. Program Distribution

"DIATH2" is currently available through the Quantum Chemistry Program Exchange, Chemistry Department, Room 204, Indiana University, Bloomington, Indiana 47401. As the remainder of programs become ready for distribution, they will also be available from QCPE.

VI. Acknowledgments

To Dr. Norman E. Sondak and the Worcester Area College Computation Center for furnishing computer time and advice on specific programming problems.

To the following WPI undergraduates for assisting on this project: David Lyons, Janet Merrill, Robert Schlain, Benjamin Thompson, and Christopher Williams.

Footnotes

1. These formulas can be found in any standard introductory quantum chemistry text. For example: L. Pauling and E. B. Wilson, Jr., "Introduction to Quantum Mechanics", McGraw-Hill Book Co., New York, 1935. pp. 326-353.
2. The specific formulas are not repeated here since they are readily available.
3. This discussion is limited to the case of the monobasic weak acid HA. However the program algorithm is capable of handling H_2A and H_3A as well.

Literature Cited

1. (a) W. Heitler and F. London, Z. Physik, 44, 455 (1927).
   (b) Y. Sugiura, Z. Physik, 45, 484 (19z7).
   (c) S. C. Wang, Phys. Rev. 31, 579 (1928).
2. (a) L. Pauling, Chem. Rev., 5, 173 (1928).
   (b) B. N. Finkelstein and G. E. Horowitz, Z. Physik, 48, 118 (1928).
3. C. A. Coulson, Trans. Farad. Soc., 33, 1479 (1937).
4. S. Weinbaum, J. Chem. Phys., 1, 591 (1933).
5. M. D. Newton, F. P. Boer, and W. N. Lipscomb, J. Am. Chem. Soc., 88, 2353 (1966).
6. Program 182

THE USE OF APL IN THE UNDERGRADUATE CHEMISTRY CURRICULUM

Thomas R. Dehner and Bruce E. Norcross
State University of New York at Binghamton
Binghamton, New York

Abstract

For the past few years members of various departments at the State University of New York at Binghamton have been exploring the applicability of APL (A Programming Language) to instructional use in the undergraduate curriculum. APL is a particularly attractive language for computer-assisted instruction because: 1) it permits direct and immediate interaction between the student and the computer; 2) the student can learn to use the language with a minimum of instruction; and 3) it not only allows programming in the conversational mode, but also has considerable computing, logical, and plotting capabilities. This paper outlines our experience with APL in the chemistry curriculum. In particular, a fairly involved program used by our students in the General Chemistry Laboratory experiment, Molar Volume of a Gas, is described in some detail. Included is a discussion of our objectives and of student reaction.

Introduction:

The past few years have witnessed a greatly increased effort to realize some of the educational potential presented by the development of computers. A number of different systems and languages have been tried in computer-aided instruction, with varying degrees of success and promise. One of the most promising languages, in our view, is APL (1), a programming language which allows direct interaction between the user and the computer by typewriter communication.

Recently a grant from the IBM Company permitted an exploration of the applicability of APL to instructional use on the college level at our institution. Since then, faculty members in chemistry, physics, biology, geology, mathematics, philosophy, and music, have taken advantage of this opportunity. Although frustrated at times with the amount of time required to develop, write, and de-bug useful programs, those involved in this exploration have for the most part remained enthusiastic about the use of computers in undergraduate teaching, and about APL as a useful language for computer-assisted instruction. In this paper some of our efforts and experiences in the use of APL in chemistry courses at the State University of New York at Binghamton will be described.

It is easy to be enthusiastic initially about using computers in teaching. The computer may indeed provide a new dimension in teaching, although care should be used in its application (2). Furthermore familiarity with that "mechanical monster" which may unwittingly send bill after bill, month after month, despite desparate pleas and claims of payment, and which has the awesome power of ruining otherwise perfect credit ratings, is certainly desirable for both teacher and student! It is also probably advantageous for students in science to become acquainted with the computer and its applications at an early stage in their training.

A variety of program types for instructional use has been suggested (3). These include:

(a) programs for data reduction and statistical treatment of experimental data;
(b) the simulated lab program, which allows the student to concentrate on the interdependence of physical variables or to examine parameter relationships in physical models;
(c) remedial drill programs or extensive tutorial programs in which the computer is the impersonal and infinitely patient instructor, allowing the student to progress without embarrassment and at his own pace through particular subject material, all the while wisely questioning the student to test his understanding of the material. This last type is undoubtedly the most difficult kind of program to write.

But, as most of you know, it's a long road from such visions to usable programs!

APL Language:

Even before defining firm objectives for efforts in CAI one must consider the computing system available and the choice of program language. Ideally, the language should be: 1) one which is easy to program and easy to use; 2) one which allows direct and immediate communication between the user and the computer; 3) one which permits extended conversational as well as computational operations; and 4) one which accommodates auxiliary interactive facilities such as slides, tapes, and CRT display. Perhaps APL, more than any other current language, comes closest to this ideal.

APL, A Programming Language (1), was first developed and used at IBM in the early 1960's. It is an interactive, algorithm-oriented time-sharing language with a single, consistent notation for many areas of discourse. By means of the typewriter terminal the user can define and store programs and execute system commands or mathematical statements, and receive an almost immediate response from the computer. The primitive functions and operations in APL offer great flexibility and simplicity in mathematical and logical operations--particularly in manipulation of vectors and matrices. Not only can the language handle fairly sophisticated calculations and mathematical operations, it is also relatively easy to program. In addition, APL has a graphic display capability in its plot functions, and allows programming in the extended conversational mode so that instructions may be given, and questions asked, in alphabetical statements as well as in mathematical formulations. Finally, programs can be easily edited and modified, and can be stored in public libraries for ready access by any user. No class time is needed for computer instruction. Brief handout instructions, supplemented by individual instruction at the terminal, have been sufficient to allow the student to use the programs which have been made available. At State University of New York at Binghamton, those students who are interested can attend a set of video-taped lectures on APL which are shown several times a year. Because of the relative simplicity of the language, a student (or faculty member) who has attended only a few of these lectures can begin to write his own programs.

We have experimented with the use of APL in the Chemistry Curriculum at every undergraduate and graduate level. Program types include laboratory simulation, laboratory data reduction, and statistical treatment of experimental data (4). We have yet to complete a tutorial-type program in chemistry, although we are currently working on one for the mole concept and stoichiometry.

Initially we were operating with an IBM 360/40 computer with a limited number of terminals. APL was available only three hours per day. Consequently, only a fairly small number of students were involved in the use of our programs. (It is interesting to note that the terminals were in use 90% of the time when APL was available, compared to 10% use during remote FORTRAN hours.) Recently we have acquired an IBM 360/67 computer, and APL is now available on campus eight hours per day. Currently we have 33 remote typewriter terminals (#2741) on campus and twenty off-campus. This year, for the first time, all of the students in general chemistry (over 300) used the computer.

General Chemistry Programs:

Lab Simulation: Initially students in one of the second semester freshman chemistry recitation sections worked with two titration programs developed by Science Research Associates (Chicago, Illinois), and somewhat modified by us. These programs allow the students to simulate acid-base titrations for a variety of weak acids, and turbidimetric titrations to obtain solubility product information for a number of slightly soluble salts.

The program TITRATE randomly chooses a pH within a specified range and asks the student to specify his initial titration parameters--i.e., molar concentration of both acid and base and volume of acid. The student then adds aliquots of base, and the computer calculates the pH after each addition. If an error is made during the course of the titration (ex.: too few points in the region of the endpoint) a reset to zero conditions is easily made, and the titration is repeated. The titration record is saved, and at the end the student can choose to see a graph of pH vs. volume of titrant for his titration. We have recently added options for first and second derivative graphs, expanded in the region of the endpoint. We plan to expand this program to include treatment of polyprotic acids and buffer solutions.

The program KSP extends the student's experience with equilibrium systems, again through simulated titration experiment. In this case the endpoint is indicated by "lasting turb'dity upon shaking", and the KSP is calculated from the titration data.

Response from the limited number of students involved with these programs was enthusiastic. (In fact several volunteers were coming at 7:00 A.M. in order to use the computer!) Although this type of program does not provide experience with lab technique, it can help the student grasp the basic concepts involved in titrimetric analysis.

Statistical Treatment of Data:

Later in the year a set of simple kinetics programs was made available to introduce the students to statistical treatment of experimental data. The student supplies the data (either from assigned problems or from laboratory experiment) together with an estimate of error in both concentration and time values. The kinetics programs apply simple first order and second order treatment to the data, calculate the best straight line fit for the data by a simple least squares treatment, and print out the rate constant calculated from the slope, and the standard deviation. Again, the student can choose to see both first and second order graphs of the data and can compare the rate constant value from the least squares treatment to the one he has obtained from his own hand-drawn graph of the same data.

An additional program in this set is used for the calculation of the activation energy from the temperature dependence of the rate constant. In this program the student can investigate the effect of error limits on the activation energy and standard deviation. Initially the computer calculates the best fit by a simple least squares method and prints out the activation energy from the slope of this line as its first approximation. The results of this calculation are then used to solve cubic equations from a more sophisticated line fitting treatment by a series of successive approximations. When successive values differ by less than 1% the calculation is terminated and the standard deviation is printed.

Again, only a small number of students used these programs, and we have as yet no valid measurement of their utility. At the very least we can say that these students were introduced early to one of the important uses of the computer for experimental data treatment, and to the fact that experimental results should always include some indication of estimated or calculated experimental error.

Laboratory Data Evaluation and Reduction:

Our most ambitious program at the freshman level is one which analyzes and stores experimental data obtained by the student in the laboratory determination of the molar volume of oxygen. The purpose of this laboratory experiment is to teach the student quantitative techniques in working with gases and to verify Avogadro's hypothesis concerning gas behavior. The student weighs out a sample of a mixture of potassium chloride (KCl) and potassium chlorate ($KClO_3$) (% Unknown) and, with appropriate apparatus which he has assembled, decomposes the $KClO_3$ by heating, and collects and measures the oxygen gas evolved. From the primary data collected (weights, volume of water displaced, temperature, barometer reading) the student can then calculate a molar volume for oxygen at standard temperature and pressure and the % $KClO_3$ in his sample.

The experimental procedure is a fairly demanding one for students at this level, and there is plenty of opportunity for experimental error. Six pieces of primary data must be obtained. On these the student performs a number of calculations, including aqueous vapor pressure and barometer scale expansion corrections, before he arrives at his final answer. Typically, a student would carry out one determination and just finish his calculations by the end of the lab period, only to find that his final answer is unacceptable and he must repeat the experiment. Clearly, it would be helpful to the student to have an immediate check on the accuracy of his primary data.

The objectives of the computer programs in the CHEMLAB workspace are:

1) To allow the student to check immediately the accuracy of his experimental measurement of the volume of oxygen;
2) To allow the student to check his calculations of molar volume and %-composition for his particular sample;
3) To store student primary data and results for subsequent analysis and grading.

The initial impetus for this program came from a paper presented at the Symposium on the Status of Computer-Assisted Instruction in Chemistry, held at Toronto in May, 1970 (5). In this paper, D. S. Olson and colleagues described the use of computer punch cards for grading chemistry laboratory reports for large classes at the Air Force Academy. We realized that by using an interactive language such as APL, and by placing computer terminals in the laboratory, we could not only ease the grading burden for our teaching assistants and collect data for class analysis, but, more importantly, could also permit the student to distinguish between experimental error and calculation error in data treatment while his apparatus was still available for repeats.

As soon as the student has finished collecting data for his first run, he goes to the computer terminal in the lab and types CHEMLAB1, which starts function execution. (Just prior to the laboratory period the lab instructor has signed onto the computer and loaded a copy of the CHEMLAB workspace from the public library into his active workspace by a simple system command. Three additional system commands initialize the program so it is ready for use by students in his lab section.) This function has three main parts. Access to each part is determined by a sequencing code in the program. The first part provides for entry of primary experimental data. The student is instructed to enter his name, sample designation, weights, temperature, barometer reading, and collected volume of oxygen. The program then calculates the theoretical volume of oxygen (using student's conditions of temperature and pressure, and using %-composition information stored in the program for the designated sample) and compares it with the student's experimental value. The computer response is determined by the agreement between these values. If the student's data is good, he is told to proceed with his calculations. His name and data are stored in the workspace and the sequencing code is advanced, so that the next time the same student calls for the program, it automatically branches to Part 2. If the agreement is fairly good, the student is allowed to proceed with his calculations, but is advised to repeat the experiment. If the agreement is poor, then the student is told that his data is unacceptable and that he must repeat the experiment. In the case of fair or poor results, leading questions are asked by the computer to help the student pinpoint likely sources of error.

Once his data is accepted by the computer, the student can then, from this primary data, calculate the weight of oxygen, the partial pressure of oxygen, and the molar volume of oxygen at STP. Part 2 of CHEMLAB1 requires the student to enter his calculated values. The program then does the same calculations, using the student's primary data which is stored in the DATA matrix and the correction factors stored in the program. The computer also calculates the theoretical values for the particular sample which the student has analyzed. The three columns of data are then printed out side by side for comparison. This allows the student to check immediately a) the validity of his calculations, and b) the accuracy of his experimental measurements. The student includes this part of the computer printout in his lab report, and is expected to discuss the agreement or disagreement of the three columns.

Again, if the student's data is acceptable he may proceed to Part 3. From the experimental weight of oxygen, the known molecular weight of O_2, the total sample weight, and the balanced equation for the reaction, he can calculate the percent potassium chlorate in his sample. He then enters the results of this calculation in Part 3. The program does the same calculations and prints out the two columns of data, along with the theoretical %-$KClO_3$ for the student's sample. Again, the student can check both the validity of his calculations and the accuracy of his results.

Once the student's data is accepted and saved by the computer, it is stored in the workspace and cannot be altered. The program LABINSTRUCT permits the laboratory instructor to retrieve all of the names and data from his lab section in a defined format for comparison, analysis, and grading.

This program was initially written in the summer of 1970 and first used by a small class in that summer session of general chemistry. The experience with the summer group indicated that the program was useful and led to fairly extensive modifications and expansion of the program. The revised program was used by over 300 students in the fall semester of 1970. Although a few students regarded the use of the computer as an extra chore, the general response of the students can be described only as enthusiastic and excited. Students suggested further modifications of the program to make it more helpful. This led to further revisions which were incorporated for use in the spring semester. Students also suggested that similar programs for other experiments would be very helpful.

The program also received strong support from the lab instructors. Usually, for this experiment, if the final calculated results are in error, the instructor must refigure each calculation until the error is discovered. It is therefore quite a chore to evaluate constructively and conscientiously 40-50 lab reports. The computer program eliminates all of the "busy work" for the instructor and allows him to pinpoint the source of error very readily. In many cases, because the student can also check his own errors, he can discuss his experimental results more intelligently.

Finally, ready access to primary data for the entire large class permitted us to examine closely the results of the experiment and to set reasonable limits of acceptability.

The CHEMLAB program has been described in some detail because it is our most successful program to date. We are encouraged by its reception and are convinced that this is a useful way to use the computer in the laboratory. The program continues to develop with each class. Future intended modifications include tutorial-type questions to guide the student into more thoughtful analysis of the experiment.

Advanced Courses:

APL programs have also been used in some of the advanced courses in the chemistry curriculum. For example, in the organic laboratory a program is used to calculate the percent composition of liquid mixtures from vapor phase chromatography data. In the introductory physical chemistry course a program has been written (by a student) which calculates the eigenvalues and wavefunctions for a one-sided potential well. By varying the parameters (depth and width of well and mass of particle) the student can follow the effect of parameter variations on the eigenvalues and attempt to discover valid relationships.

The program for determination of activation energy, mentioned earlier, was actually developed for use by juniors and seniors in a physical chemistry lab experiment on the internal rotation of amides, measured by a nuclear magnetic resonance method. The data obtained are not susceptible to simple least squares treatment, so the students use a computer program to calculate weighting functions for statistical treatment of their NMR data (6).

Finally, in the physical-organic courses, students have been required to develop their own line-fitting program in APL for treatment of kinetics and other experimental data.

Other Departments:

Several other departments at State University of New York at Binghamton have also been experimenting with APL in their courses. In the genetics and biostatistics courses fairly simple programs have proven to be both fascinating and useful to the student (7). The program RANDOMDRIFT is an example of a computer-simulated experiment in genetics which could not possibly be done by the students in the laboratory. If a population is limited in size, or if only a small fraction of the population is responsible for the offspring of the next generation, the genetic composition of the population will fluctuate in a random manner. The smaller the population, the greater the random fluctuation. This is known as the Sewall-Wright Effect. Instead of merely being informed of this experimental observation, the student is permitted to experience the sensation of discovery by using an APL function which is programmed so that half of the population in each generation is randomly selected to be parents for the next generation. The student starts with four allelic genes, selects the size of the population sample and continues the reproductive process until three of the four loci are lost from the population by random drift.

Extended tutorial programs which simulate physical laws otherwise difficult for the student to investigate have been most successful in the physics department (8). Prior to the simulated experiment a series of questions in multiple choice format familiarizes the student with the problem and provides a guide to intelligent selection of parameters. Wrong choices lead to appropriate branches in the program which contain additional drill and guidance. The student specifies all important parameters for the experiment, and observes the results, usually in graphical form. Additional tutorial questions guide the student's analysis of the results and help him discover the parameter relationships. For example, the program ORBIT plots the orbit of a satellite launched tangentially from a central body. The student specifies the central body and its mass, the satellite mass, and initial distance and velocity. He then observes the effects of varying these parameters on the orbit of the satellite. Other tutorials in physics include an investigation of wave behavior and interference patterns, and analysis of Michelson-Morley type experiments to arrive at postulates of relativity. Indeed, these programs have been so successful that the physics department this year has replaced the introductory physics laboratory with a "structured combination of computer laboratory and hands-on demonstrations". The traditional introductory lab has been upgraded and moved to the sophomore year.

Conclusion:

Although our experience so far is insufficient to make extensive judgements about the educational value of our efforts in computer-assisted instruction, there are a number of very encouraging aspects of our exploration to date:

1) We have developed some programs which appear to be useful and unique as instructional aids and which can be successfully integrated into our course programs. The more workable model programs we have on hand, the easier it is to formulate and develop additional programs.
2) We are succeeding in introducing a significant number of students to the computer and its potential uses, and they are responding enthusiastically.
3) A significant number of students, on their own intiative, are writing their own programs in APL and continuing to use the computer for course work, lab data treatment, and research data analysis.

These are the encouraging factors. The discouraging ones are the cost in both time and money to support and develop such a program. These cost factors are difficult to determine, since these programs undergo almost continual evolution even after considerable use. Recent conservative estimates at our institution (8) indicate that the development of a long tutorial like ORBIT, or an involved program like CHEMLAB, requires approximately 100-120 hours of full time effort. Thus the cost for author's time would be in the range of $1000 for such a program. The only other cost to the department in the development of the program is CPU time, at $100/hour. In most cases this is small compared to the cost of the author's time.

The major costs in a CAI program are terminal rental, at roughly $1500 per year per terminal, and student-generated CPU time. A typical calculation (assuming five terminals, an average student connect time of an hour a week, and an average 20:1 ratio of connect time to CPU time) yields an estimated cost of $20,000 per year for a class of 100 students, or roughly $200 per student. Obviously, if the cost remains at this level, one must decide whether it is more useful to have the computer available or to hire another faculty member (or two)!

At present however, given the necessary funds, a willing and able computer center staff and the continued interest of both faculty and students, we plan to continue to work on the development of programs for computer-assisted instruction.

Acknowledgements:

The authors wish to thank the staff of the computer center at State University of New York at Binghamton for their cooperation and assistance. We are particularly indebted to Mr. James Higgins, academic manager of the computer center; to Mr. Kevin Kelly, who did most of the initial programming for CHEMLAB, and to Mrs. Anne Kellerman who has helped with the revision of CHEMLAB. We also gratefully aknowledge the financial support of the IBM Corporation.

REFERENCES

1. K. Iverson, A Programming Language, John Wiley and Sons, Inc., New York. 1962; K. Iverson and A. Falkoff, APL/360 Users Manual, International Business Machines, New York, 1968; P. Berry, APL/360 Primer, International Business Machines, New York, 1970.

2. J. A. Young, J. Chem. Educ., 47:11, 758 (1970).

3. (a) "Computers in Undergraduate Education: Mathematics, Physics, Statistics and Chemistry", proceedings of Conference at University of Maryland, December 8-9, 1967.
   (b) R. C. Atkinson and H. A. Wilson, "Computers in Chemical Education", Science, 162, 73 (1968).
   (c) "The March on Armonk", proceedings of the APL Users Conference, State University of New York at Binghamton, July 11-12, 1969.
   (d) F. D. Tabbutt, "Computers in Chemical Education", Chem. & Eng. News, January 19, 1970, p. 44.
   (e) Computer Instruction Issue, J. Chem. Educ., 47:2 (1970).
   (f) Symposium on the Status of Computer-Assisted Instruction in Chemistry, Joint Conference of the Chemical Institute of Canada with the American Chemical Society, Toronto, Canada, May 27, 1970.

4. T. R. Dehner and B. E. Norcross, "The Use of APL (A Programming Language) in Computer-Assisted Instruction in Freshman Chemistry", presented at 158th National American Chemical Society Meeting, New York, September 8, 1969.

5. D. S. Olson, S. R. Carroll, L. A. King, J. F. Altenburg, Symposium on the Status of Computer-Assisted Instruction in Chemistry, Joint Conference of the Chemical Institute of Canada with the American Chemical Society, Toronto, Canada, May 27, 1970.

6. B. E. Norcross, R. H. Cartmell, G. Hall, "The Hindered Rotation of Amides", presented at 158th National American Chemical Society Meeting, New York, September 8, 1969.

7. A. P. Mueller, J. Biol. Ed., 3:2, 159 (1969); Proceedings of the APL Users Conference, State University of New York at Binghamton, July 11-12, 1969.

8. C. Stannard, Am. J. of Physics, 38:12, 1416 (1970).

Appendix

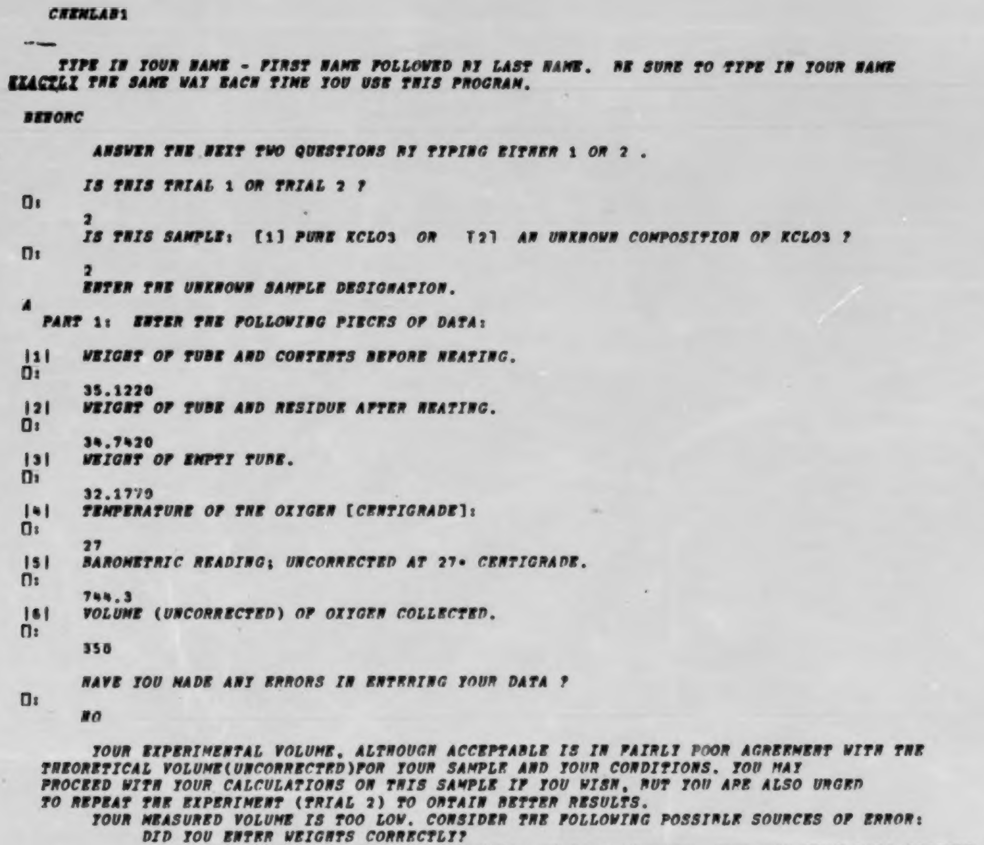
Figure Number	Title
1	Beginning portion of APL program, CHEMLAB1.
2	Portion of APL program, CHEMLAB1.
3	Sample execution of CHEMLAB1, part 1.
4	Sample execution of CHEMLAB1, part 2.
5	Sample execution of CHEMLAB1, part 3.
6	Sample printout of stored data from CHEMLAB 1, by program INSTRUCTOR.

FIGURE 1

```
      ∇CHEMLAB1[⎕]∇
    ∇ CHEMLAB1
[1]     ''
[2]     ''
[3]   START:'        TYPE IN YOUR NAME - FIRST NAME FOLLOWED BY LAST NAME.  BE SURE TO TYPE IN YOUR NAME'
[4]     '  EXACTLY THE SAME WAY EACH TIME YOU USE THIS PROGRAM.'
[5]     ''
[6]     YES←1+NO←I←0
[7]     INPUT←((⌊/(INPUT=' ')/⍳15)-1)⌽INPUT←15↑⍞,15⍴' '
[8]   ON:→(((I<1↑⍴DATA)∧((~IO)×~ID)),(IO←×/INPUT=NAME[I;]),(ID←×/NAME[I←I+1;]∊' '))/ON,MATCH,FILL
[9]     'NAME TABLE FILLED.'
[10]    →0
[11]  FILL:NAME[I;]←INPUT
[12]    ''
[13]    '        WHEN YOU ENTER DATA, ALL WEIGHTS SHOULD BE IN GRAMS, ALL VOLUMES IN MILLILITERS, AND'
[14]    ' ALL PRESSURES IN MILLIMETERS[OF HG], UNLESS SPECIFIED OTHERWISE.
[15]    ''
[16]    '        DO NOT INCLUDE UNITS WHEN ENTERING DATA.'
[17]    ''
[18]    '       DO YOU NEED ADDITIONAL INSTRUCTIONS?'
[19]    →(~⎕)/MATCH
[20]    ''
[21]    '         DATA IS ENTERED BY TYPING IN THE DESIRED CHARACTERS AND PRESSING THE ''RETURN'' KEY.'
[22]    '  IF A MISTAKE IS MADE IN TYPING, DO NOT PRESS THE ''RETURN'' KEY; USE THE ''BACKSPACE'' KEY TO'
[23]    '  BACKSPACE THE PRINTING ELEMENT UNTIL IT IS POSITIONED DIRECTLY OVER THE FIRST (ON LEFT) FAULTY'
[24]    '  CHARACTER.  DEPRESS THE ''ATTN'' KEY AND REPLACE THE FAULTY CHARACTER(S) BY TYPING IN THE CORRECT'
[25]    '  CHARACTER(S).  RETYPE ALL CHARACTERS TO THE RIGHT OF THE CHARACTER(S) REPLACED.  PRESS THE ''RETURN'' KEY.'
[26]    ''
[27]    '         IF A MISTAKE HAS ALREADY BEEN ENTERED BY PRESSING ''RETURN'', CONTINUE ON UNTIL I ASK IF ANY'
[28]    '  MISTAKES HAVE BEEN MADE.'
[29]    ''
[30]  MATCH:''
[31]    '         ANSWER THE NEXT TWO QUESTIONS BY TYPING EITHER 1 OR 2 .'
[32]    ''
[33]    '       IS THIS TRIAL 1 OR TRIAL 2 ?'
[34]    SWITCH←⎕
[35]    '       IS THIS SAMPLE:  [1] PURE KCLO3  OR   [2]  AN UNKNOWN COMPOSITION OF KCLO3 ?'
[36]    NUMBAH←1
[37]    →(⎕=2)/GOON
[38]    '       ENTER THE UNKNOWN SAMPLE DESIGNATION.'
[39]    NUMBAH←NUMBAH+'ABCDEFGHIJKL'⍳1↑(((A=' ')/⍳10)-1)⌽A←10↑⍞,9⍴' '
[40]  GOON:DATA[I;SWITCH;1]←NUMBAH
[41]    →((⍳4)∊DATA[I;SWITCH;15])/FAR,FURTHER,FINALE,FAREST
[42]  FAREST:'        MY RECORDS INDICATE YOU HAVE ALREADY ENTERED VALID INFORMATION FOR THE SAMPLE'
```

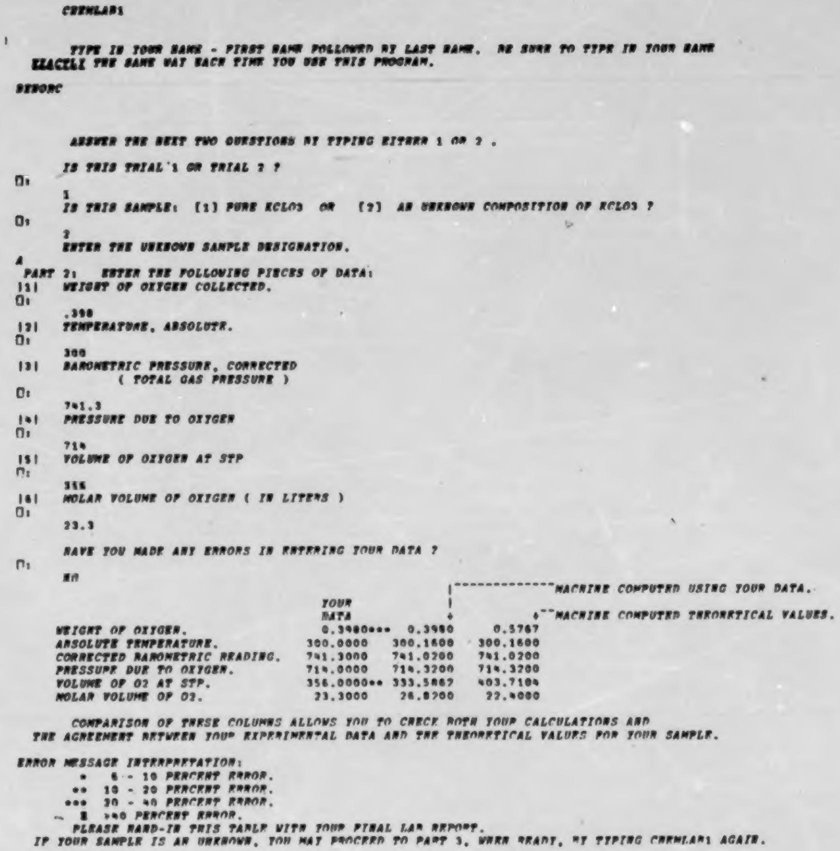
```
[58]  '|3|    WEIGHT OF EMPTY TUBE.'
[59]  DATA[I;SWITCH;4]←⎕
[60]  →(⌶26)+FUDGY
[61]  RETURN←RETURN,(⌶26)+1
[62]  '|4|    TEMPERATURE OF THE OXYGEN [CENTIGRADE]:'
[63]  DATA[I;SWITCH;5]←⌊TEMP←⎕
[64]  →(0.5>TEMP-DATA[I;SWITCH;5])/AROUND
[65]  DATA[I;SWITCH;5]←DATA[I;SWITCH;5]+1
[66] AROUND:→(⌶26)+FUDGY
[67]  RETURN←RETURN,(⌶26)+1
[68]  '|5|    BAROMETRIC READING; UNCORRECTED AT ';DATA[I;SWITCH;5];'° CENTI GRADE.'
[69]  DATA[I;SWITCH;8]←⎕
[70]  →(⌶26)+FUDGY
[71]  RETURN←RETURN,(⌶26)+1
[72]  '|6|    VOLUME (UNCORRECTED) OF OXYGEN COLLECTED.'
[73]  DATA[I;SWITCH;7]←⎕
[74]  →(⌶26)+FUDGY
[75]  RETURN←RETURN,LAST←(⌶26)+1
[76]  ''
[77]  '       HAVE YOU MADE ANY ERRORS IN ENTERING YOUR DATA ?'
[78]  →(~⎕)/(PTR=⍳3)/GOTCHA,GOAL,GOLLY
[79]  ''
[80]  '       ENTER THE NUMBER OR NUMBERS ( FOUND IN THE | | ) OF THE INPUT STATEMENTS ASSOCIATED WITH'
[81]  '  THE DATA YOU WISH TO CORRECT.  IF MORE THAN ONE NUMBER IS TO BE ENTERED, SEPARATE WITH BLANKS.'
[82]  GOTO←0,(((⍳(DQ←⍴RETURN)-1)∊⎕),1)/RETURN
[83]  GATE←(2,DQ)⍴RETURN,1⌽(RETURN-2)
[84]  →((⍴GOTO)≤2)/RETURN[DQ]
[85]  GOTO←1↓GOTO
[86]  FUDGY←(⌶26)-GATE[2;(GATE[1;]∊GOTO[1])/⍳DQ]+2
[87]  →GOTO[1]
[88] GOTCHA:PART←((20×⌊0.5+DATA[I;SWITCH;8]÷20)-620)÷20
[89]  PART←DATA[I;SWITCH;8]-TCTABLE[DATA[I;SWITCH;5]-15;PART]÷0.03
[90]  PART←PART-VPTABLE[DATA[I;SWITCH;5]-15]
[91]  DATA[I;SWITCH;6]←(PERCENT[NUMBAH]×(DATA[I;SWITCH;2]-DATA[I;SWITCH;4])×25528020×(DATA[I;SWITCH;5]+
      273.16))
[92]  DATA[I;SWITCH;6]←DATA[I;SWITCH;6]÷33475.758×PART
[93]  FUDGE←|(DATA[I;SWITCH;6]-DATA[I;SWITCH;7])÷DATA[I;SWITCH;6]
[94]  ''
[95]  →(FUDGE<0.1,((0.1≤FUDGE)∧(FUDGE≤0.2)),((0.2≤FUDGE)⍲(FUDGE≤0.3)),(FUDGE>0.3))/VG,RG,PG,UA
[96] VG:'       YOUR EXPERIMENTAL VOLUME IS IN VERY GOOD AGREEMENT WITH THE THEORETICAL VOLUME(UNCORRECTED)'
[97]  '  FOR YOUR SAMPLE AND YOUR EXPERIMENTAL CONDITIONS. PROCEED WITH YOUR CALCULATIONS.'
[98]  →SMUDGE
```

FIGURE 2



```
   CHEMLAB1

  --
    TYPE IN YOUR NAME - FIRST NAME FOLLOWED BY LAST NAME.  BE SURE TO TYPE IN YOUR NAME
EXACTLY THE SAME WAY EACH TIME YOU USE THIS PROGRAM.

  BENORC

        ANSWER THE NEXT TWO QUESTIONS BY TYPING EITHER 1 OR 2 .

      IS THIS TRIAL 1 OR TRIAL 2 ?
⎕:
      2
      IS THIS SAMPLE:  [1] PURE KCLO3  OR   [2]  AN UNKNOWN COMPOSITION OF KCLO3 ?
⎕:
      2
      ENTER THE UNKNOWN SAMPLE DESIGNATION.
A
  PART 1:  ENTER THE FOLLOWING PIECES OF DATA:

|1|   WEIGHT OF TUBE AND CONTENTS BEFORE HEATING.
⎕:
      35.1220
|2|   WEIGHT OF TUBE AND RESIDUE AFTER HEATING.
⎕:
      34.7420
|3|   WEIGHT OF EMPTY TUBE.
⎕:
      32.1779
|4|   TEMPERATURE OF THE OXYGEN [CENTIGRADE]:
⎕:
      27
|5|   BAROMETRIC READING; UNCORRECTED AT 27° CENTIGRADE.
⎕:
      744.3
|6|   VOLUME (UNCORRECTED) OF OXYGEN COLLECTED.
⎕:
      350

      HAVE YOU MADE ANY ERRORS IN ENTERING YOUR DATA ?
⎕:
      NO

       YOUR EXPERIMENTAL VOLUME, ALTHOUGH ACCEPTABLE IS IN FAIRLY POOR AGREEMENT WITH THE
  THEORETICAL VOLUME(UNCORRECTED)FOR YOUR SAMPLE AND YOUR CONDITIONS. YOU MAY
  PROCEED WITH YOUR CALCULATIONS ON THIS SAMPLE IF YOU WISH, BUT YOU ARE ALSO URGED
  TO REPEAT THE EXPERIMENT (TRIAL 2) TO OBTAIN BETTER RESULTS.
       YOUR MEASURED VOLUME IS TOO LOW. CONSIDER THE FOLLOWING POSSIBLE SOURCES OF ERROR:
             DID YOU ENTER WEIGHTS CORRECTLY?
             DID YOU COMPLETELY DECOMPOSE THE KCLO3 IN YOUR SAMPLE?  (DID YOU HEAT LONG ENOUGH)
             DID YOU HAVE AN AIRTIGHT SYSTEM (NO LEAKS!)?
      WHEN YOU NEED ME AGAIN, JUST CALL (CHEMLAB1)!
```

FIGURE 3



```
     CHEMLAB1

      TYPE IN YOUR NAME - FIRST NAME FOLLOWED BY LAST NAME.  BE SURE TO TYPE IN YOUR NAME
  EXACTLY THE SAME WAY EACH TIME YOU USE THIS PROGRAM.

BENORC



      ANSWER THE NEXT TWO QUESTIONS BY TYPING EITHER 1 OR 2 .

     IS THIS TRIAL 1 OR TRIAL 2 ?
□:
     1
     IS THIS SAMPLE:  [1] PURE KCLO3  OR   [2]  AN UNKNOWN COMPOSITION OF KCLO3 ?
□:
     2
     ENTER THE UNKNOWN SAMPLE DESIGNATION.
A
 PART 2:   ENTER THE FOLLOWING PIECES OF DATA:
|1|   WEIGHT OF OXYGEN COLLECTED.
□:
     .398
|2|   TEMPERATURE, ABSOLUTE.
□:
     300
|3|   BAROMETRIC PRESSURE, CORRECTED
             ( TOTAL GAS PRESSURE )
□:
     741.3
|4|   PRESSURE DUE TO OXYGEN
□:
     714
|5|   VOLUME OF OXYGEN AT STP
□:
     356
|6|   MOLAR VOLUME OF OXYGEN ( IN LITERS )
□:
     23.3

     HAVE YOU MADE ANY ERRORS IN ENTERING YOUR DATA ?
□:
     NO
                                               |-------------MACHINE COMPUTED USING YOUR DATA.
                                  YOUR         |
                                  DATA         +            +--MACHINE COMPUTED THEORETICAL VALUES.
   WEIGHT OF OXYGEN.              0.3980***  0.3980     0.5767
   ABSOLUTE TEMPERATURE.        300.0000   300.1600   300.1600
   CORRECTED BAROMETRIC READING. 741.3000   741.0200   741.0200
   PRESSURE DUE TO OXYGEN.      714.0000   714.3200   714.3200
   VOLUME OF O2 AT STP.         356.0000** 333.5867   403.7104
   MOLAR VOLUME OF O2.           23.3000    26.8200    22.4000

      COMPARISON OF THESE COLUMNS ALLOWS YOU TO CHECK BOTH YOUR CALCULATIONS AND
  THE AGREEMENT BETWEEN YOUR EXPERIMENTAL DATA AND THE THEORETICAL VALUES FOR YOUR SAMPLE.

ERROR MESSAGE INTERPRETATION:
       *    5 - 10 PERCENT ERROR.
      **  10 - 20 PERCENT ERROR.
     ***  20 - 40 PERCENT ERROR.
    ~  ■  >40 PERCENT ERROR.
      PLEASE HAND-IN THIS TABLE WITH YOUR FINAL LAB REPORT.
  IF YOUR SAMPLE IS AN UNKNOWN, YOU MAY PROCEED TO PART 3, WHEN READY, BY TYPING CHEMLAB1 AGAIN.
```

FIGURE 4

```
      CHEMLAB1


       TYPE IN YOUR NAME - FIRST NAME FOLLOWED BY LAST NAME.  BE SURE TO TYPE IN YOUR NAME
  EXACTLY THE SAME WAY EACH TIME YOU USE THIS PROGRAM.

TOMDEHN

       ANSWER THE NEXT TWO QUESTIONS BY TYPING EITHER 1 OR 2 .

      IS THIS TRIAL 1 OR TRIAL 2 ?
□:
      1
      IS THIS SAMPLE:  (1) PURE KCLO3  OR   (2)  AN UNKNOWN COMPOSITION OF KCLO3 ?
□:
      2
      ENTER THE UNKNOWN SAMPLE DESIGNATION.
A
  PART 3:  ENTER THE FOLLOWING DATA:
|1|   MOLES OF OXYGEN.
□:
      .165
|2|   MOLES OF POTASSIUM CHLORATE DECOMPOSED.
□:
      .110
|3|   WEIGHT OF POTASSIUM CHLORATE IN SAMPLE.
□:
      .145
|4|   WEIGHT OF SAMPLE USED.
□:
      2.94
|5|   PERCENT OF POTASSIUM CHLORATE IN SAMPLE.
□:
      51

      HAVE YOU MADE ANY ERRORS IN ENTERING YOUR DATA ?
□:
      YES

      ENTER THE NUMBER OR NUMBERS ( FOUND IN THE | | ) OF THE INPUT STATEMENTS ASSOCIATED WITH
  THE DATA YOU WISH TO CORRECT.  IF MORE THAN ONE NUMBER IS TO BE ENTERED, SEPARATE WITH BLANKS.
□:
      3
|3|   WEIGHT OF POTASSIUM CHLORATE IN SAMPLE.
□:
      1.45



      HAVE YOU MADE ANY ERRORS IN ENTERING YOUR DATA ?
□:
      NO

                                              YOUR          |"""MACHINE COMPUTED USING YOUR DATA..
                                              DATA          *

  MOLES OF OXYGEN EVOLVED.                    0.1650     0.0178
  MOLS OF KCLO3 DECOMPOSED.                   0.1100     0.0117


  WEIGHT OF KCLO3 IN SAMPLE.                  1.4500     1.4350
  WEIGHT OF SAMPLE USED.                      2.9400     2.9450
  PERCENT KCLO3 IN SAMPLE.                   51.0000    48.7280

  THEORETICAL PERCENT POTASSIUM CHLORATE FOR YOUR SAMPLE IS 56 PERCENT.
   YOU HAVE FINISHED COMPUTER TREATMENT OF DATA FOR THIS SAMPLE, AND MAY NOT ALTER
 ANY OF THE DATA.  PLEASE HAND IN THIS TABLE WITH YOUR REPORT.  IF YOU HAVE MADE ANY
 ERRORS IN YOUR FINAL CALCULATIONS, CORRECT THEM AND INCLUDE THE CORRECTIONS IN
 YOUR LAB REPORT.  YOUR REPORT SHOULD INCLUDE A DISCUSSION OF YOUR EXPERIMENTAL ERROR.
```

FIGURE 5

```
INSTRUCTOR

[I]:          TUBE AND            TUBE AND          WEIGHT OF        OXYGEN            UNCORRECTED
         CONTENTS BEFORE           RESIDUE          EMPTY TUBE     TEMPERATURE          BAROMETRIC
              HEATING          AFTER HEATING                                            PRESSURE

[II]:        ABSOLUTE            CORRECTED        PRESSURE DUE     VOLUME OF         MOLAR VOLUME
            TEMPERATURE          BAROMETRIC        TO OXYGEN       OXYGEN AT         OF O2 AT STP
                                  READING                             STP

[III]:        STUDENT VOLUME             COMPUTED VOLUME
            OF O2:UNCORRECTED          OF O2:UNCORRECTED

STUDENT:  BENORC

[I]:
SAMPLE:B           35.1220          34.7240          32.1770          27.0000          744.3000
SAMPLE:A           35.1220          34.7420          32.1770        0 27.0000          744.3000

[II]:
SAMPLE:B          300.0000         741.3000         714.0000         356.0000           23.3000
SAMPLE:A          300.0000         741.0000         714.0000         400.0000           22.8000

[III]:
SAMPLE:B          390.0000         471.8479
SAMPLE:A          350.0000         471.8479

STUDENT:  TOMDERN

[I]:
SAMPLE:A           35.1220          34.5600          32.1770          27.0000          744.3000
SAMPLE:NONE         0.0000           0.0000           0.0000           0.0000            0.0000

[II]:
SAMPLE:A          300.0000         741.0000         714.3000         401.0000           22.4000
SAMPLE:NONE         0.0000           0.0000           0.0000           0.0000            0.0000

[III]:
SAMPLE:A          468.0000         471.8479
SAMPLE:NONE         0.0000           0.0000
```

FIGURE 6

COMPUTERS AND CHEMICAL EDUCATION AT PITT

K. Jeffrey Johnson
University of Pittsburgh
Pittsburgh, Pennsylvania 15213

During the past two and one-half years I have been implementing computing techniques where appropriate in the chemistry curriculum. This effort includes five areas of emphasis: tutorial-drill CAI in freshman chemistry, data reduction and simulation programs, teaching programming in physical chemistry, offering an application course in FORTRAN programming and numerical analysis (1) and use of an interactive graphics system (2). Most of the computing is done on an IBM 360/50, using the Pitt Time-Sharing System, PTSS (3). The interactive graphics facility utilizes the University of California at Santa Barbara Culler-Fried On-Line System (2).

CAI in Freshman Chemistry

PTSS supports the teacher oriented CAI language, CATALYST/PIL (4). CATALYST is a language that facilitates I/O and branching. PIL is a JOSS derived interpretive language designed for short, interactive student jobs (5). The CATALYST/PIL interface provides a nearly ideal language for CAI.

One of the most attractive features of this interface is the ease with which students can make chemical calculations. Two of the traditional stumbling blocks of freshman chemistry students are stoichiometry and aqueous equilibrium. Most of our CAI programs include numerical problems in these areas. They involve the use of PIL generated random numbers. The students use the arithmetic features of PIL to solve the numerical problems.

Since the fall term of 1970, the students enrolled in our general chemistry courses have been given a memo describing PTSS and the CAI programs. Little or no lecture time is allotted for discussion of PTSS or use of the programs. Student usage of these programs is entirely voluntary. At this writing the memo describes 16 programs. Two are not numerical: a review of periodic properties, and a lesson on acid-base concepts. There are five stoichiometry lessons, two on gas law calculations, five on aqueous equilibrium problems, one on Faraday's laws and one on colligative properties.

At any point the student may type "help", and be provided a hint. The following example will illustrate this feature along with the random number and the arithmetic features. The computer dialog is represented in the upper case:

```
EXERCISE:  THE IONIC SOLID MX2 DISSOCIATES AS FOLLOWS:

          MX2 <===>  M(++) + 2X(-)

IF THE SOLUBILITY PRODUCT KSP = 6.409E-16
CALCULATE THE SOLUBILITY OF MX2 IN PURE WATER

>help
WANT SOME HELP AT THIS POINT ?

LET Z EQUAL THE SOLUBILITY OF MX2 IN MOLES/LITER.
THEN IT FOLLOWS THAT IN THE SATURATED SOLUTION:

          (M(++)) = Z MOLES/LITER
          (X(-)) = 2*Z MOLES/LITER

AND (M(++))*(X(-))**2 = KSP

NOW TRY THE PROBLEM AGAIN.

>(6.409E-16/4)**(1/3)

RIGHT.  NOW, ...
```

The student feedback has been encouraging. They are asking for more lessons. Results from questionnaires indicate that the students feel their time is well spent and that CAI is an effective teaching tool. Predictably, the students most enthusiastic about CAI are the A and B students. Yet the emphasis is on tutorial-drill lessons for average and below average students. The latter group of students "can never find the time" to locate a terminal and try out the programs. It can only be hoped that with increased availability of terminals and with the development of a more complete and effective library of CAI programs, these motivational problems will be solved.

Data Reduction and Simulation Programs

For some purposes it is worthwhile to provide students with a library of black-box FORTRAN programs. Many of the interesting simulation and data reduction problems involve coding that is either beyond the programming expertise of the student or would be too time consuming and diverting from the chemistry involved. For example, in our sophomore level analytical chemistry course, the students are given an optional problem set that refers to a library of FORTRAN programs. One of these is to calculate the potential of a metal-metal ion electrode in aqueous ammoniacal solution as a function of pH. The problem involves the synthesis of three concepts: acid-base equilibria, complexation equilibria and application of the Nernst equation. A pedestrian use of this program would be for verification of hand-calculated values. The more exciting feature is that good students now have a tool to provide answers to the question, "what if?" In this case, the interested student can investigate the pH behavior of a series of metal electrodes (e.g., Zn, Cd, Hg), and attempt to explain the results.

One of the programs made available to students in Physical Chemistry solves the particle in a finite box problem. Here the adjustable parameters include the mass of the particle, height of the potential well and dimensions of the box. The eigenvalues are the roots of the transcendental equation:

$$\tan\left\{\frac{2a(2mE)^{\frac{1}{2}}}{\hbar}\right\} = \frac{2(V_o-E)^{\frac{1}{2}}E^{\frac{1}{2}}}{2E-V_o}$$

where m is the mass of the particle, 2a is the length of the box, V_o is the potential height and E is the energy of particle. By increasing Vo the student can observe the approach to the infinite potential solution, and by increasing m and/or a, he can observe the approach to a classical continuum of energy levels.

This application of computers in chemical education provides a tremendous challenge to educators. We must come up with challenging, stimulating and pedagogically useful problems. We now have a teaching tool that will add a new dimension to chemical education if we use it properly.

PIL in Physical Chemistry

In the winter term of 1971, the physical chemistry laboratory course was modified to include an hour recitation each week. I was asked to present in one lecture enough PIL so that students could write their own linear regression analysis programs. I compiled an eight page handout describing PTSS and PIL and containing several sample programs. The assignment that followed the lecture was to write a linear least squares program with the following output: intercept and slope, standard deviations of intercept and slope, correlation coefficient, and a table containing the independent and dependent variables, the calculated ordinates and the error associated with each ordinate. The program was due in two weeks. One of the handout sheets contained all the required formulas, some sample data with results, and some debugging information. For those students with some programming experience, this approach is satisfactory. It represents a rather challenging assignment for those who lack programming experience. It is the ease of learning and using PIL that makes this approach realistic.

Numerical Methods in Chemistry--A Computer Application Course

This upper-division undergraduate/graduate course has been described elsewhere (1). The following description is taken from the Pitt College of Arts and Sciences bulletin:

> A computer applications course for chemists. Topics include programming in the PIL and FORTRAN languages, solutions to chemical data-reduction problems, techniques in solving linear, nonlinear and differential systems of equations, determination of eigenvalues and eigenvectors, Huckel MO analysis, NMR spectral analysis, nonlinear least-squares analysis, and, as time allows, several special topics. Emphasis is on presentation and application rather than derivation and discussion. One of the main objectives is to enable the student to adapt one of several published programs to his own use. Prerequisite: Mathematics 24 (Calculus)

The student feedback from this course has been very gratifying. The students appreciate the condenstaion of the relevant information contained in several math and computer science courses into an applications course for chemists.

Interactive Graphics

Pitt is privileged to be one of the Universities in the On-Line System network centered at the University of California at Santa Barbrra. Our terminal consists of the two-level keyboard, storage oscilloscope, write-only teletype and X-Y plotter. This interactive graphics facility represents a quantum jump forward in utilization of computers in chemical education. Typically the student will use this facility to simulate some physical system. For example, he may be interested in high-resolution NMR. Let's assume he wants to simulate ABX spectra. The adjustable parameters include three chemical shifts, three coupling constants, the line shape and the line width for each of the 14 peaks. Closed form expressions for the 14 frequencies and intensities are available (6). The coding to calculate the 14 peaks and the total calculated spectrum is very quickly learned by good undergraduate students. So, in short order he can generate a family of ABX spectra and discover the effects of subtle changes in coupling constants, chemical shifts, relaxation times, line shapes, etc.

During the fall term of 1970, I issued a memo to all students enrolled in physical chemistry courses. I invited those with QPA of 3.0 or better to sign up for a special one credit course. The course met one evening a week. Eight students enrolled. After two lecture hours the students were able to start programming. The only requirement was a well-documented project due at the end of the term. Most of the students chose their own projects. They included: a simulation of the kinetic system $A \rightleftarrows B \rightleftarrows C$, a study of the solubility behavior of silver halides as a function of halide concentration, analysis of the Michaelis-Menten equation and Lineweaver-Burk plots of simple enzyme inhibition systems, and a simulation of the Monod-Wyman-Changeux model of enzyme cooperativity. One of the students was so hooked on the system that he wanted more information about the NSF proposal. He was going to graduate school and wanted to get a terminal.

I interested one of my graduate students in making film loops. We are preparing several ten minute film loops on orbital contours, NMR spectra, titrations, kinetic schemes, complex formation curves, etc.

There is no doubt in my mind that interactive graphics is the way to go. For most purposes one graph is worth ten thousand numbers. When we can no longer afford a private line to Santa Barbara, we will develop an interactive graphics system here at Pitt.

Concluding Remarks

From this experience I have a number of comments regarding the use of computers in chemical education. First, it is not everybody's bag. I recommend that the use of computers be voluntary at all levels, from CAI to interactive graphics. Even with a reasonably reliable time-sharing system, there are a number of time-consuming and sometimes irritating obstacles. There is a chronic shortage of terminals during prime time. There is the system command language to be learned. In IBM FORTRAN, there is the rigid input formatting problem. And every now and then the system crashes. Many students can handle all this with equanimity. I think those who cannot should be spared.

For some students a little knowledge about computers is a dangerous thing indeed. They get a malicious chuckle out of sabotaging a program or crashing the system. For example, in the freshman CAI effort, it was decided for convenience to give the entire class the same project number and password. Soon someone learned how to change the password. Until the computer center devised a protection for the password, the entire class was at this student's mercy. Then someone learned how to modify and destroy data sets. To my embarrassment the computer was coming up with strange and sometimes shocking statements.

I am afraid some students would have learned more chemistry if there were no diverting computer. The PIL language is delightfully easy to learn. Students quickly find out about demonstration games like blackjack, tic-tac-toe, etc. Some forget about the freshman chemistry CAI lessons.

I see in all this two promising applications of computers in chemical education. One is to provide a teaching tool to students to make their learning experience a little easier and more interesting. The other application is more exciting: provide the really good students with a tool to solve some challenging problems. Let them take on problems that go far beyond those at the end of the chapter. We must not use the computer for the sake of using the computer. I think that with care we can use computers to make learning chemistry more interesting and our teaching more effective.

Acknowledgements

Support of this research from the National Science Foundation Grants GJ-696 and GU-3184 is gratefully acknowledged. I also wish to thank the many programmers, undergraduate and graduate students who have contributed to this project, particularly Ellene Tratras (CAI), Dennis Doerfler (FORTRAN) and Susanna Levitt (interactive graphics).

REFERENCES

1. K. Jeffrey Johnson, J. Chem. Ed., 47, 819 (1970).

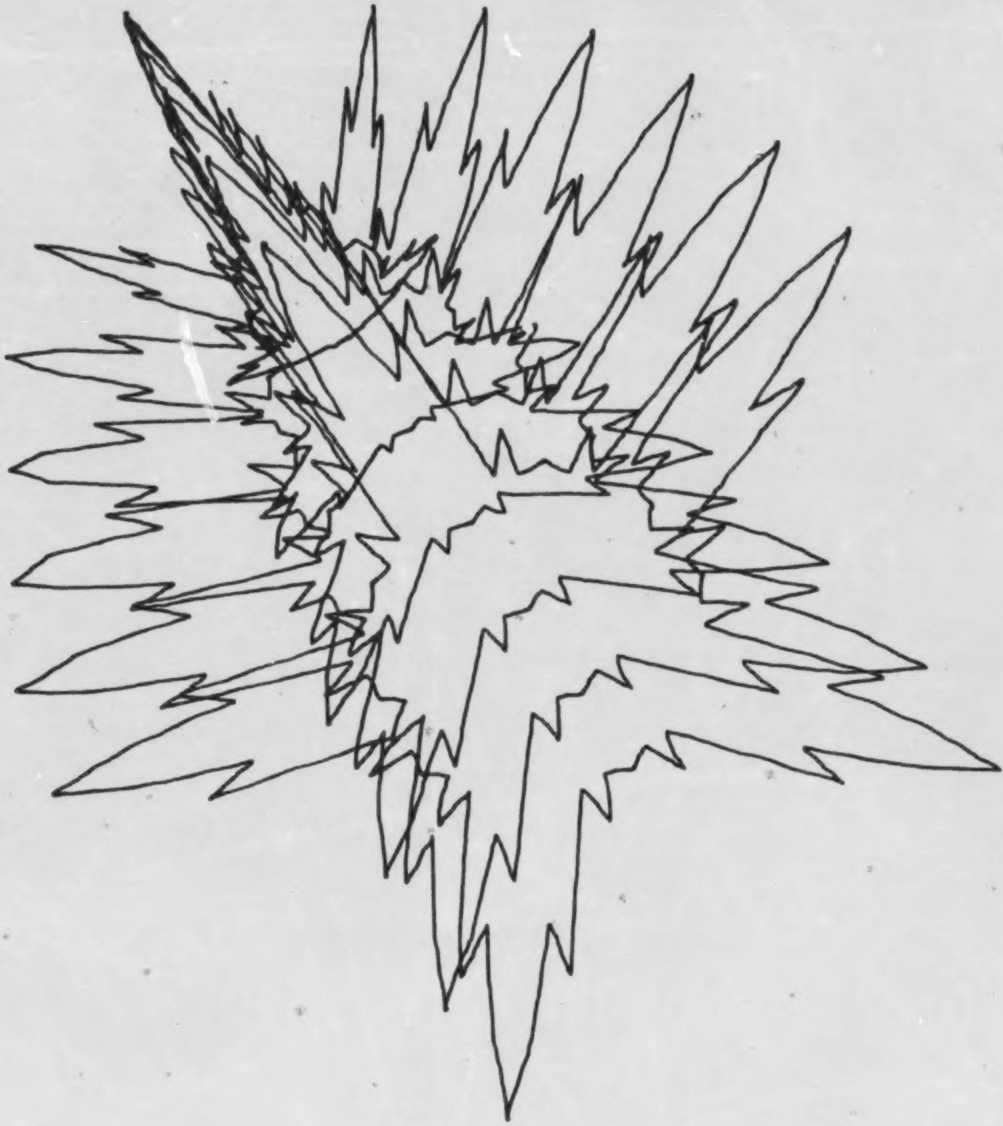
2. Carl S. Ewig, J. Thomas Gerig and David O. Harris, J. Chem. Ed., 47, 97 (1970).

3. G. Badger, E. Johnson and R. Philips, "The Pitt Time-Sharing System for the IBM System 360: Two Years Experience," Proceedings of the Fall Joint Computer Conference, San Francisco, December, 1968.

4. Thomas Dwyer, "A Lesson Designer's Guide to CATALYST and the CATALYST/PIL Interface", Computer Center, University of Pittsburgh, October, 1969.

5. "PIL/L - Pitt Interpretive Language for the IBM System/360 Model 50", Computer Center, University of Pittsburgh, 1968.

6. J. A. Pople, W. G. Schneider and H. J. Bernstein, "High-resolution Nuclear Magnetic Resonance", McGraw Hill, 1959; p. 132 ff.



Stephen C. Harvey

A SURVEY OF APPLICATION SOFTWARE FOR SOCIAL DATA ANALYSIS INSTRUCTION

Ronald E. Anderson
University of Minnesota
Minneapolis, Minnesota 55455

Social science software packages have been proposed, promoted, described, and even surveyed (Meyers, 1969), yet no author has undertaken the project of examining them as a whole and drawing conclusions on the entire range of social software developments. I undertook this challenge to analyze the broad picture, realizing that in a short essay I could never cover all of the important points or even cover a few points in adequate depth. My goal will have been achieved if several broad issues are pointed out such that discussion increases, and progress in instructional software is thereby stimulated.

BACKGROUND

The social research requirements of the U.S. census stimulated the invention of punch card machines in the late 1800's. Likewise the complex analysis demands of the social science research of the mid-1900's continues to stimulate the creation of many computer programs for statistical data analysis. In this recent demand for statistical processing, the social and behavioral sciences share much in common with other fields, especially biomedicine. Although many application packages for statistical data analysis were designed and developed by social scientists, the earliest package to receive wide-spread distribution came from the biomedical computing group at UCLA (Dixon, 1967). The package, now called "BMD", is still the most widely distributed statistical library. Many other programs and systems have used the BMD package as a starting point, elaborating on specific programs within the package.

The BMD program collection originated a little over 10 years ago, but within only 10 years there are hundreds of statistical library packages and thousands of miscellaneous individual "canned" special-purpose statistical programs.

TERMINOLOGY

A collection of programs maintained as a unit is called a library; if the programs are closely related, e.g., all statistical, then the library is called a package. If a package of programs is tightly integrated such that output from one program is automatically input to another program, then it is frequently called a system or a software system. Strictly speaking, it is an applications software system as opposed to a general software system, which is not tied to the needs of a particular field. Associated with each social application system is an explicit or implicit language. Utilizing Sammet's (1969) language categories, these are application-oriented languages. The language specifies the way instructions are structured in order to obtain the desired processing. With some of the more recent systems such as DATA-TEXT and OMNITAB, it is common to call the system a "language". The newer, time-sharing systems are not generally referred to as languages because so much of the communication takes place in a question-answering dialogue.

INVENTORY OF SOFTWARE SYSTEMS

Taking an inventory of programs is such an arduous task that few have tackled the project and none have carried it far enough. Currently an organization at the University of Wisconsin, called the National Program Library and Central Program Inventory Service for the Social Sciences (NPL/CPIS), under the direction of James R. Taylor, has begun to serve as a central information clearing house. Such efforts hopefully will solve some of our information and program dissemination problems.

Table 1 lists a few of the numerous packages available for social data analysis. The program packages are selected for inclusion primarily because they are widely available or representative of a common program type. Numerous programs have been omitted from this list, but their omission in no way implies negative evaluation.

The historical progression of program construction over the past ten years is one of increasing complexity and power. Prior to 1960 only miscellaneous, single-purpose programs were written. Usually they were written in machine language, and little attempt was made to run a program on more than one machine. The BMD packaging operation in the early 1960's began the trend to create libraries and packages. Probably the first integrated system was the Multivariate Statistical Analyzer (MSA) of Jones (1965), which was based upon the Cooley and Lohnes book, Multivariate Procedures for the Behavioral Sciences.

It was not until the mid-1960's that language-oriented systems emerged, even though specifications for a statistical language were given as early as 1960 (Douglas and Mitchell). OMNITAB, built at the National Bureau of Standards, became popular with government statisticians in the mid-1960's, but DATA-TEXT was the first widely circulated language-oriented system for social data analysis.

As the 1970's begin, several other systems are also installed in many different locations. University-generated systems such as SPSS (Statistical Package for the Social Sciences) have spread widely across the country. On the commercial side, CROSSTABS has diffused into over 100 establishments, and is advertised by Cambridge Computer Associates as the most widely used commercial statistical reporting package. CROSSTABS borrows heavily in concept and language from DATA-TEXT.

The DATA-TEXT system has undergone extensive reprogramming, language extensions, and in addition has received the blessing of a well-written language text (Armor and Couch, 1971). It is my guess that DATA-TEXT will become the most popular social data analysis system of the early 1970's.

The current batch-oriented systems, while suffering from certain weaknesses, are a vast improvement over the situation of only a half decade ago. The foremost improvement is in "user-orientation", which means the system and its documentation are tailored toward simple, convenient utilization by non-programmers as well as programmers. Armor (1970) analyzed the common features of three language-oriented systems and found them to include the following features: 1) file management operations and high degree of user control over input and output, 2) liberal labeling and built-in codebooks, 3) large variety of transformational statements available, 4) wide range of statistical computations possible, and 5) facility to select specified cases, e.g., missing data removal. Not only do these systems provide power and flexibility, but since instructions are prepared according to a language syntax, errors in preparation are less likely and easier to track down.

A quick glance at Table 1 reveals that most software packages are non-commercial and batch-oriented. There are relatively few systems listed in the commercial category. Any manufacturer-designed system is considered commercial, e.g., IBM's Scientific Subroutine Package (SSP), even though they are sometimes distributed without fee. Thus "commercial" identifies the primary goal of the sponsoring organization. Some systems such as SPSS and DATA-TEXT charge exportation fees in order to maintain the programs, but these are still considered non-commercial.

Most systems were developed in university environments, and commercial institutions have taken them for their own consumption. The only area in which commercial interests have taken the lead is with programs designed for market research. Programs), where social scientists have worked closely with commercial software groups to develop a useful social science system.

One dimension not clarified by the categories of Table 1 is the computer size to which the system is oriented. Significantly, almost all systems were developed on large computers and require a large amount of central memory. Those packages oriented toward a medium-size computer include DAFPACK, PSH (Pacific State Hospital), TABLE-GEN, CROSSTABS, SSIP (Social Sciences Instructional Programming Project), and SAP. A number of these were designed by small liberal arts schools. Interestingly enough, none of the packages except perhaps SAP, are mini-computer oriented. This vacuum is probably due to the difficulty in squeezing much software capability into a tiny computer, but, in addition, mini-computers have only recently become fairly wide-spread. Several groups, e.g. Carleton College and the University of Minnesota Sociology Department, are writing packages in the FOCAL language. Creative developments might lead to a social science oriented system for mini-computers that could bring computing to many social science students located in schools which cannot afford more than a mini-computer.

TIME-SHARING

Batch-oriented systems are still more common than conversational systems, as indicated by

Table 1. Other than the fact that most computing is still done in a batch mode, there are several additional factors that account for the slow growth of social science time-sharing systems:

1. There seems to be a partial return to the special-purpose program for conversational modes. With on-line operating systems providing so many data management and retrieval functions, the large application system is less essential. For instance, such operations as case extraction and error correction in data files can usually be done easily with a system text editor.
2. For some time to come most users will probably use a combination of batch and conversational modes. We are still experimenting with time-sharing interactional situations to discover what problems benefit from conversational computing. Nygreen (1971) recently described path analysis program which clearly benefits from its interactive capability. We are far from understanding the significance of time-sharing for social science research and instruction, and it may be that new modes of computing emerge, e.g., "Instant Turnaround" as found at Penn State University, that add important alternative modes to consider.
3. Time-sharing brought chaos to program designers because there is little standardization, expenses are high, and the future is very uncertain. Thus there are a great many obstacles for anyone brave enough to consider developing a software system in a time-sharing environment.

Recently there has been a growing consideration of how time-sharing potentialities can best be utilized for instruction. Cline and Meyers (1970) suggest that a desirable future state of affairs is the student of sociology interacting with a data analysis system. What is also needed is research assessing the effects of alternate computing modes upon the character and quality of instruction.

In some ways the challenge of time-sharing has been beneficial; much more debate and research occurs now over the design of meaningful application software systems. Given the nature of data analysis (Tukey, 1969; and Muller, 1969) as an exploratory and repetitious enterprise, conversational computing will certainly have an important role. It holds the promise of bringing the researcher into closer contact with his data during the analysis process. And it holds the promise of providing students with a powerful learning situation in which quick exploration can be done along alternate paths chosen by the student in an ongoing process of interaction with the program system. Some of the programs developed by James Sullivan at the Economics Laboratory at University of California, Santa Barbara, are good illustrations of the possibilities of this approach.

STATE-OF-THE-ART CONCLUSIONS

1. The systems with the best design and greatest potential are not yet generally available. Systems such as BEAST (Brookings Economic and Statistical Translator) and TRACE III are not yet finished, and when they are completed they will not be easily exportable. (Perhaps these systems are aiming so high that they never will become reality.)
2. Almost every system presumes that the data are structured in a simple rectangular manner. The only provisions for handling complex data structures, e.g., hierarchical structures, remain outside the data analysis system (see e.g., Ellis and Nelson, 1970). Although some complex structure manipulation can be done with LANSOR and its predecessor, NWAY, the only systems to thoroughly integrate the capability are BEAST (Kidd, 1969) and TRACE III (Cooperband, 1971). Although some would argue that this omission is trivial, their argument makes the situation even more serious. It is serious because there are many instances in which complex data structure capability is needed, yet since our software systems do not have it we tend to define the problem such that the more complex structural analysis is avoided. When the attributes of computer programs shape the character of research, it is indeed a serious matter.
3. Very few of the systems provide simple addition of user-supplied routines, e.g., FORTRAN subroutines, without a significant amount of effort. The LANSOR system, because it is a pre-compiler, allows users to intermix FORTRAN and LANSOR statements. Systems that provide absolutely no provision for user-supplied extensions are insults to their users.

4. At the present time, statistical software systems are too narrow, neglecting possible linkages into such computer tasks as content analysis and simulation. For instance, it might be very useful to shift automatically from a word-counting text routine to a multivariate statistical operation. Computer systems should never be obstacles to the adoption of new techniques and analysis programs.

5. The computational accuracy of standard statistical routines has been put into question by recent research (Longley, 1967). Greater attention needs to be paid to this problem and students should be thoroughly apprised of likely problems due to round-off errors.

CONCLUSIONS SPECIFIC TO INSTRUCTION

1. A variety of packages and programs should be available in a well-rounded instructional environment where the goal is social research training. No single system satisfies all social research needs, and it is not obvious that standardization of social science application languages would at this early stage be desirable. We do not yet know how inclusive a single system can be while maintaining reasonable levels of efficiency and user-orientation. Without user-orientation the teaching problem is insurmountable.

2. Importing, installing, and implementing a package tends to be a very costly, difficult challenge. Acquiring a new program package and loading it onto the computer system is usually not overwhelming, but implementing it adequately requires a large staff of patient, persistent programmer/teachers. Even if the programs are completely debugged and the users' manual is completely unambiguous, someone must prepare exercises and integrate program use with the substance of the courses. These tasks consume time and expense, and since resources are not always available, it is not surprising that powerful packages frequently remain untouched by the very people who need them most.

3. From a sociological perspective, the key to success with instructional computing applications is the campus computer subculture. On each campus there are likely to be a core of computer-men and -women who set the tone for computing and largely determine the emergence of norms concerning when and how to get involved with computers. The status and occupational positions of members of the hard-core computer club is much more important than computer science has ever recognized. Should we not be doing research on such questions as what difference does it make when deans rather than laboratory technicians are the campus computer elite?

   Cline (1968) identified several user types; the entrepreneur who hires his computer work out; the dependent user who overburdens computer consultants; the co-opted applied mathematician who forsakes his social research for a life with computers; and the scorner who looks down on the computer whirlwinds from the humanist tower. Studies should be done to find out how persons filling these various roles differ in their research activities and what impact the computer systems have upon their work. One might ask such questions as "Does a BMD campus differ from a DATA-TEXT campus or an IMPRESS campus in the number of dependent user types typically found?"

4. One extremely significant consequence of the wide availability of user-oriented systems is the large increase in the segment of computer users who are not trained in programming or even elementary computing. These users relate to the computer only as a user of a canned program or system; they must rely totally upon program documentation and any consultants they can find to struggle through their computer project. Software system users' manuals have long implied such an approach to computing, but finally this year a book was published which focuses entirely upon how to use social science software systems (Blackman and Goldstein, 1971). The Blackman and Goldstein book concentrates almost entirely upon use of P-STAT, but the dangerous implication of the little text is that the student can get along just fine without any additional understanding of computing.

   Package use by itself does not provide much instructional value. Instructors should not teach package usage unless:

   (a) Package usage is taught in non-credit, short-course mode;

158

(b) or package usage is an integral part of the learning of a specific substantive concept; and a good example of this desired integration is given in a paper by James Davis (this conference);

(c) or package usage is taught simultaneously with computing concepts and programming techniques.

These suggestions are made for undergraduate education. Although they apply equally to graduate and post-graduate education, often the resistance to learn a little computer science is so great among these older groups that there is little chance to change the situation.

One approach which could solve many of these problems with untrained computer users is that of simultaneously teaching statistics and programming (e.g. Bulgren, 1971). Programming can be taught in laboratories while statistics are taught in lectures. Exercises are given that build upon the programming skills to make the statistics concept easier to understand. Likewise, having the statistics problems to work on stimulates interest in learning programming.

SUMMARY

Over the past ten years we have truly been flooded with programs and systems for social data analysis. The better systems are powerful, user-oriented, and well-documented. Yet a number of significant problems remain in most if not all existing systems. What really deserves systematic investigation is the social and instructional consequences of the new wave of software packages.

TABLE 1

APPLICATION-ORIENTED SOFTWARE SYSTEMS FOR SOCIAL DATA ANALYSIS

	BATCH		TIME-SHARING	
	Non-commercial	Commercial	Non-commercial	Commercial
STATISTICAL LIBRARIES	BMD MSP UMSTAT	SSP	SSIP	ITS-STAT
INTEGRATED STATISTICAL PACKAGES OR SYSTEMS	MSA BC-TRY OSIRIS TSAR DAFPACK PSH ASCOP ARIEL SSUPAC STATJOB BEAST OMNITAB SPSS DATA-TEXT P-STAT		IMPRESS TRACE III APL STATPACK DATANAL DATAX TROLL	SAP REACT BASIS
TABULATION-ORIENTED SYSTEMS	LANSOR NUCROS TABLE-MAKER TABLE-GEN	CROSSTABS QUESTAIRE		QPAK
DATA MANAGEMENT OR TEXT-ORIENTED SYSTEMS	SIPS GRASS CODAS BEAR GI	QWICK QWERY PROFILE UTILITY-CODER	ADMINS GIPSY	

Note: A glossary of terms and acronyms is available from the author.

REFERENCES

Anderson, Ronald E.
1969 "Introduction to Lansor (Language for Social Research)", Sociology Data Center, Department of Sociology, University of Minnesota, Minneapolis, Minnesota.

Armor, David J.
1970 "Developments in Data Analysis Systems for the Social Sciences". Social Science Information, 3 (June): 145-156.

Armor, David J. and Arthur S. Couch
1971 *The Data-Text Primer*. Free Press (to appear, Spring 1971).

Blackman, Sheldon and Kenneth M. Goldstein
1971 "An Intro to Data Management in the Behavioral and Social Sciences", Wiley.

Bulgren, William
1971 "A Computer-Assisted Approach to Elementary Statistics", Wadsworth Publishing Co., Belmont, California.

Cline, Hugh F.
1968 "Computer Instruction for Scholars in the Humanities", Computers and the Humanities, 3 (September): 31-40.

Cline, Hugh F. and Edmund Meyers, Jr.
1970 "Problem-Solving Computer Systems for Instruction in Sociology", American Sociologist, 5 (Nov.): 365-70.

Cooperband, Avlin S., Wm. Moore, R. Meeker, G. Shure
1971 "TRACE III: An Implicit Programming System for Inductive Data Analysis", Center for Computer-Based Behavioral Studies, University of California, Los Angeles.

Dixon, W. J.
1967 "BMD - Biomedical Computer Programs", Berkeley and Los Angeles: University of California Press.

Douglas, A. S. and A. J. Mitchell
1970 "A Data Description Language for Hierarchical Data Files", TP-11, Data and Computation Center for the Social Sciences, University of Wisconsin, Madison.

Jones, Kenneth J.
1965 "The Multivariate Statistical Analyzer", Behavioral Science 10, 326-327.

Kidd, Stephen W.
1969 "Incorporating Complex Data Structures into a Language for Social Science", Proceedings of the Fall Joint Computer Conference, 1969: 453-461.

Longley, James W.
1967 "An Appraisal of Least Squares Programs for the Electronic Computer from the Viewpoint of the User", American Statistical Association Journal, 62(September): 819-841.

Meyers, Edmund D., Jr.
1969 "Survey of Social Science Computing Systems", Project IMPRESS, Dartmouth College, Hanover, N. H.

Muller, Mervin E.
1969 "Statistics and Computers in Relation to Large Data Bases", In Milton and Nelder (eds) "Statistical Computation", Academic Press, New York.

Nygreen, G. T.
1971 "Interactive Path Analysis", American Sociologist 6(February): 37-43.

Sammet, Jean E.
1969 "Programming Languages: History and Fundamentals", Prentice-Hall.

Tukey, John W.
1969 "Analyzing Data: Sanctification or Detective Work?", American Psychologist 24, 2(February): 83-91.

Note: A complete bibliography may be obtained from the author at the Department of Sociology, University of Minnesota.

INSTANT TURNAROUND AND CONVERSATIONAL COMPUTING AS INSTRUCTIONAL TOOLS IN THE SOCIAL SCIENCES

Francis M. Sim, Laurence S. Rosen, and Glen D. Kreider
The Pennsylvania State University
University Park, Pennsylvania

In general, there have been two approaches to the use of computational facilities in undergraduate instruction--batch processing systems and conversational or "time-shared" systems. There are several disadvantages in the use of batch computing systems as instructional tools in undergraduate social science courses, especially courses in research methods and data analysis. (Cline and Meyers, 1970; Kreider, *et al.*, 1970; Meyers, 1969a; Sim, *et al.*, 1970; Sim, *et al.*, 1971; Williams, *et al.*, 1970) On the other hand, conversational systems are quite useful as teaching aids in such courses from both the intructors' and the students' points of view, because of their ease of use, speedy turnaround, and versatility, viz., orientation towards use by the inexperienced student as well as by the experienced programmer.

There is a third type of computing system which, although essentially a batch processing system, exhibits several of those features of conversational systems considered useful for instruction in social science data analysis. (Kreider, *et al.*, 1970; Sim, *et al.*, 1971; Williams, *et al.*, 1970) Because this type of system yields output immediately, it generally is referred to as a fast-batch or an instant turnaround system.

As both conversational and *instant* turnaround batch systems exhibit somewhat similar features, there are merits of each as an instructional tool in the undergraduate curriculum. Furthermore, as both systems may be set up on a network basis, the question of relative merit is as important to the small college or university with a limited budget and/or limited computing facilities as it is to large universities. It is the purpose of this paper to explore and compare the conversational approach and the instant turnaround approach for social science instruction in data analysis. In doing so we shall rely upon examples from two particular systems with which we are somewhat familar, the IMPRESS (time-sharing) system at Dartmouth College and the WIT (instant turnaround) system at The Pennsylvania State University.

IMPRESS

> "Time-sharing" means that many people can use the computer simultaneously--or seem to, at least. Even though the computer can answer only one command at a time, it works so fast that dozens of people typing at separate terminals will all get their answers in a matter of seconds. (Lawrence, 1970, pp.6-7)

The key characteristic of time-sharing is that it is a conversational system. This means that once the user has "signed-on" to the system the computer will interact with the user by responding to commands and, when completed, will "ask" for further instructions until the user terminates the session. For example, after the user "signs-on", the system responds by "asking" the user what sort of work he wishes to do. The user then responds and, if the command is within the system's repertory, the system acts upon the command.

Without entering into a detailed discussion of technical complexities, we may state that the speed and mass simultaneous use (or availability) characteristic of time-sharing systems can be explained in terms of three of the fundamental aspects of any computational system: time allotment (scheduling), task orientation, and communications.

Systems such as Dartmouth's have earned the sobriquet "time-shared" because, for example, after giving a brief period of CPU time to user A the computer will shift its attention to user B who, after a similar period, will be shunted aside in favor of user C, and so on until user A is again "at bat." Available CPU time, which is a finite commodity, is shared with several users at once. In order to take advantage of a system organized in such a manner, users are expected to view their jobs as a series of small discrete tasks, or microtasks, rather than single but complete tasks (macrotasks), as is characteristic of batch processers.

> A macrotask is that part of a user's work that is conceptually significant by itself, even though it may not complete the computational portion of his intellectual endeavors. Roughly, it corresponds to a "job" as that word is used in most computer installations. On the other hand, a microtask is some small part of input preparation, say a single line of code, that is about as small a unit as could be meaningfully responded to by the system being used. (Sim, et al., 1971, p. 16)

Each user command to a time-sharing system is a microtask regardless of whether that command is itself a one word response to a system generated question about a library program parameter or a subsequent command to run that program. Conceptually, neither corresponds, from the user's viewpoint, to a whole complete job. Because of the manner in which CPU time is alloted to each user, many of the allowable tasks are deliberately simple operations, or microtasks, which may be completed in a single pass through the CPU. Other tasks, such as file maintenance, are structured so as to by-pass the CPU altogether. Response to such tasks is quick, if not instantaneous. A program, however, cannot avoid being channeled through the CPU in order to compile and/or run. If a user's program does not run to completion during the user's allotted CPU time the job must wait until accessed by the CPU as many times as is necessary to complete the run. But "whole program" execution may be made to seem as quick as other types of operations, and this leads us to our final category, communications.

Communications between the user and the computer in a time-sharing environment is by means of a teletype or some other typewriter terminal. The printing rate is ten or fifteen characters per second. (Special terminals designed for time-sharing achieve a rate of thirty characters per second, but few time-sharing systems are as yet equipped to handle the faster transmission rate.) As large quantities of data may be produced quickly within a time-sharing environment, it is almost inevitable that transmission of output will become backed-up because of the comparative slowness of one-character-at-a-time output devices. Rather than being a fault, however, the resultant continuous printing of output satisfies the user's demand for instantaneous turnaround and serves to mask, and perhaps eliminate the frustrations of, the non-productive wait-time between successive passes through the CPU.

Project IMPRESS (Interdisciplinary Machine Processing for Research and Education in the Social Sciences) is a package of approximately thirty inter-related programs (written in BASIC or time-sharing Fortran) which is resident in Dartmouth's G.E. 635 time-sharing system. IMPRESS is an interactive, teaching oriented, system which, because of its stored data capabilities and its statistical routines, is not limited to use solely by inexperienced and unsophisticated undergraduates. (Cline and Meyers, 1970; Meyers, 1969a; Meyers, 1969b) However, actual use of the IMPRESS system by students is relatively simple. Each student enrolled in a course which uses the time-sharing system is assigned an account number which validates the student for use of the system, including IMPRESS. After the student signs-on the system and requests use of IMPRESS, the system will interact with the student by "asking" questions regarding what the student wishes to do, viz., what data to work with, what mapping procedure to be used, what cross-tabulations to be produced, and so on. If the student is a novice he may request instructions and/or further information at any step. (Cline and Meyers, 1970) If mistakes are made or if problems arise which the student cannot rectify, the student may start again with little loss of time. (Meyers, 1970a)

To summarize:

> We wanted undergraduates to have ready access to the wealth of existing social science data in such a fashion that minimal constraints would be imposed upon the manner in which these data could be examined....IMPRESS utilizes the computer to perform the drudgery associated with data analysis so that the student can be free to pursue his own substantive questions.... (Meyers, 1970b, p. 8.23)

WIT (Watfor Instant Turnaround)

The term "instant turnaround" refers to a batch processing system which provides the user with almost immediate response regardless of the type of output requested--program listing, error diagnostics, or program results. However, response is in the batch mode--to the user's whole job rather than line-by-line as in a time-sharing environment. The throughput speed of such a system depends on the "self-service" character of job submission as well as on restrictions regarding input, output, and CPU time for any one job. Such a system is enhanced through storage of programs and data on direct access memory devices. (Kreider, et al., 1970, p. 8.16)

Further illustration of the instant turnaround approach can be made through discussion of the system organization and hardware employed by the WIT system at Penn State. The main research and instructional computing facility at Penn State consists of a single IBM 360/67 along with its many and varied peripheral devices. Partitioning of both the high-speed and low-speed core storage areas allows simultaneous execution of several standard batch, conversational, remote entry (RJE), and WIT jobs. (Sim, et al., 1971; Williams, et al., 1970)

The partition assigned to WIT consists of 128,000 bytes of core storage, and the WATFOR compiler (a superset of Fortran) is resident in the partition. On the University Park campus, four minimum IBM 360/20's with line printer and card reader serve as dedicated terminal devices for the partition and are restricted to instructional use. They are housed in two locations remote from the Computation Center Building, with an attendant present during scheduled hours. The card reader and printer are both medium speed devices--five hundred cards per minute input and three hundred lines per minute output. (Kreider, et al., 1970; Sim, et al., 1971; Williams, et al., 1970)

As stated previously, the speed of the WIT system depends upon restrictions on the use of the system. The most important restrictions are in regard to input, output, and CPU time. Jobs entering the WIT partition are allowed a maximum of five seconds CPU time, and output is limited to a maximum of six hundred lines. Input is somewhat limited since excessive card reading will cut into the user's available run time, but it is possible to process more than five hundred cards in a run. As a result of these restrictions, the WIT system can provide almost immediate turnaround within a batch processing mode.

In general, only students are allowed access at the WIT terminals. Faculty wishing to enter jobs into the WATFOR partition for non-instructional purposes may do so only at other terminals of the system. Jobs entered from the student terminals are given very high priority.

SUPERTAB is a tabular data analysis program written in WATFOR and is resident in the library dedicated to the WIT partition. The program was designed for use in teaching of an elementary course in social research methodology. (Sim, et al., 1971) Although the program is simple enough to be used by a novice, it produces all tabulations necessary for a three-variable "survey elaboration". (Rosenberg, 1968)

Use of the student terminals for entry to the WIT system (whether for specific use of SUPERTAB or not) is simple and is easily learned. Each student enrolled in any course which uses WIT receives a single "ticket" or job card which is used in all runs the student submits. The job card is placed in front of the student's deck along with whatever control, and/or program, and/or data cards are necessary. The student then loads the deck into the terminal read hopper and presses the "start" button. By the time the deck has been read and retrieved from the stacker the printer often has begun printing the output. The student then walks five feet to the printer and removes his output. The total elapsed time is thirty to forty-five seconds. Heavy usage periods, such as during the last week or two of the term, do not increase time spent at the terminal (time spent waiting for use of the terminal and turnaround time at the terminal) to more than about five or six minutes. If errors need to be corrected or if changes are desired, the student may resubmit his run for another try whenever he is ready to do so.

To summarize:

> The present system, then, provides students with the ability to access the computer an almost unlimited number of times in a term on an "instant demand" basis with a turnaround of less than a minute apart from the really peak end-of-term periods. It lets the user test out his program, provides diagnostics (in English, incidentally) for the whole job, and allows work on improvement of output, all with relatively small amounts of attention by the CPU. (Williams, et al., 1970, pp.11.4-11.5; Note that this quotation corrects an omission error on pages 11.4 and 11.5 in Proceedings of a Conference on Computers in the Undergraduate Curricula, 1970.)

Common Arguments for the Superiority of Conversational Computing

It is likely that many social scientists who are familiar with electronic computation would argue that any computational system that does not "interact" with the user is less than optimal for instructional needs. It seems to be fairly commonly believed that "interactive" or "conversational" computer systems are uniformly better for instructional purposes than are "batch processors", and that only tradition, avoidance of novelty, and disposition to choose a "satisfying" solution that is already well known over a new and unfamiliar (though better) way, and perhaps ignorance, holds back widespread demand for conversational computing. These beliefs are often coupled with the argument that while such "real third generation" computer systems cost more in terms of hard dollars, they have benefits that have not been adequately cost accounted in improvements of instruction; that they have instructional consequences such that a student's learning rate increases by some order of magnitude over what can be achieved with batch processing; that once these costs become properly assessed, and effective "third generation" systems become widely available, there will be a large shift to such "time-sharing" environments for instruction.

To the best of our knowledge, arguments for conversational computing and against batch processing in instruction are not well developed and documented on a point-by-point basis, in part we would suppose because of the assumption that they are mostly self-evident, except for the point above about the deceptiveness of manifest dollar costs. As we understand the matter, the conversationist argument may be stated in the following interrelated points. (1) In a conversational system, the user (particularly the student) does not need to learn and/or remember as much to get started and keep going in his work. There seem to be two ideas here: in a conversational system the user can be prompted and guided through his job in a step-by-step way, he will not have to remember any card formats or as many arbitrary conventions about what comes where, and the system can be programmed to make sure that all of the formally relevant information is supplied by the user before execution; and in addition to this, statistical package batch processors are typically heavy on computer jargon that is unknown and unnecessary to the novice (or to the expert user either for that matter). (2) By one or another contiguity principle of behavior, immediate response improves the user's ability to respond in turn efficiently and effectively. Particularly, for instructional purposes, immediate response is more effective as a corrective and reinforcement than is delayed response. (3) The final main point, again, is that while a time-sharing job typically costs more in bookkeeping terms, it is also worth more, because of the kinds of advantages conferred under the first and second points.

With reference to conventional batch processing, it would be difficult to refute the above points. However, we believe that several of the assumptions upon which the above points are based need to be questioned, especially in light of the fact that the WIT system *is* a batch system which does not have the characteristics generally attributed to all batch processing systems.

The first of these is the almost explicit assumption that only time-sharing avoids long turnaround times (on parameter evaluations, if not on analysis executions). The system that we have described gives immediate response to the user's job though it does not give response to lines of specifications as he types them at the keypunch. At the least this raises the question as to whether it is the timing of the response or its level that provides sufficient contiguity for the desired effects. The second assumption seems even more complicated by received practice in computation, and it is that batch processing must be oriented to research and thus to use by experts. Certainly, any conversational system is inherently tutorial, or at least it leads the programmer into a teaching mode, even though it does not absolutely require this of him. Since it is based on interaction on small units of information at a time, the system must present inself for new input when it has digested the last "line." Since it is indicating readiness, it might as well show what it is ready for; and just in case the user doesn't know what that is, the system might as well be able to give him some clues on how to find out, if not provide a complete explanation. But is it true that a batch processor must be anti-tutorial? We believe that it is more true to say that in the past many such systems have been more difficult to learn about than is minimally necessary because they have assumed that there may be long turnaround times, that these make it advisable to get as much as one can on a single submission, and that anyway the user will use the program over a sufficient period of time to become facile and thus amortize his sunk costs. WIT is a system where these assumptions are wrong. It is one which is considerably less difficult to learn about than many batch processors, and it may not be any more difficult or less effective to use than conversationally based programs.

Comparative Features of IMPRESS and SUPERTAB

Unfortunately, it is difficult to adjudicate any of these questions, since there are few well conceived and executed studies of batch processing versus time-sharing (Gold, 1969; Schatzoff, et al., 1967), and no really relevant ones of time-sharing in comparison to an instant turnaround batch processor. There are, however, several papers that report on the Dartmouth IMPRESS system, principally authored by Professor Edmund D. Meyers, Jr., and we can use them as a basis for some remarks that we can hope will be propaedeutic to sounder analysis of these systems. (Cline and Meyers, 1970; Meyers, 1969b; Meyers, 1970a; Meyers, 1970b) The observations will only touch on what seem to us to be some salient and instructive comparative features.

"Technicality" of Use

In a recent statement on instructional computing, based on IMPRESS, it is asserted that,

> ". . . the technical difficulties of getting introductory students to use batch computers and job control languages have usually been overwhelming; time spent learning technical matters of computing is stolen from time which should be spent on substantive issues in sociology.
>
> For instructional purposes - where immediate feedback and reinforcement are key elements - interactive computer systems are ideal." (Cline and Meyers, 1970)

Among other things, it is implied that only an interactive system can escape being so laden with technical demands that the novitiate is distracted from the real business at hand; their subsequent remarks lay the source of these demands in the same place as our earlier ones, viz., the orientation of batch systems to research. However, we do not agree that interaction is either sufficient or necessary to escape this bane for the student.

Interactive systems can fall prey to technicalese that debilitates them for elementary (non-programming) instruction. A good illustration of this can be found in the ADMINS system. It is an interactive system built for social research, and one that is especially good for pre-analysis data processing activities, e.g., codebook preparation, data generation, editing, and the like. Of course, it is just these sorts of applications that we feel ought to be elided from early instruction in research methods; the motivation of such topics depends on seeing larger analysis issues simply and clearly first. In a recent comparative review of several interactive systems, Professor Meyers has praised the ADMINS system for its virtues, but also scored it because ". . . it does not appear to be user-oriented." (Meyers, 1970a, p. 159) Our point is an obvious one, but it is easy to miss when enthusiastic; interactive computing is tutorial in character, but it is easy to teach the wrong subject to the wrong student if one's attention is elsewhere. In fact, all systems seem to require a certain minimum amount of knowledge that is irrelevant to the user's substantive interests, and interactive ones do not eliminate this.

Let us now try to show that SUPERTAB is not inherently more technical than an interactive program by making some comparisons of details between it and IMPRESS. Examples of IMPRESS jobs are given in each of the references cited in the preceding note, although all capabilities of the system, especially some of the more recently implemented ones, are not illustrated in these examples. Nevertheless, the jobs illustrated in the sources, as well as jobs illustrating other features (such as mapping and sorting) which Professor Meyers has kindly made available to us, can be executed using SUPERTAB on the WIT system. As the data files used in both cases are quite similar, the correspondence is rather good.

To begin with, one customarily associates interactive systems with relatively mnemonic language features--for variable reference, for processing commands, for statistical option declarations. Variable reference in IMPRESS is accomplished with ease. During parameter development the user must declare which variables will be selected into his private file from the survey that he is using, and then later he must specify which of his file variables will be used in particular tabulations. To do this he uses "names" of up to six characters that were assigned to the variables when the survey was placed "on line", i.e., set up for direct access retrieval by the system. No variable numbers or other arbitrary designations are used, at least the designations are not arbitrary within the ability of the librarian to devise such brief names with real descriptive power. Full labels and descriptions of the variables are in the codebook for the survey, which is not stored on line, but presumably is quickly available to the user. The user will also get a longer title for the variable on output of his results, and he can get this printed on his console by request after he has set up his file.

SUPERTAB does not differ greatly from this. The student declares which variables he wishes to use by entering somewhat descriptive eight character mnemonic devices on his "VARIABLES" card. SUPERTAB differs from IMPRESS, however, in that the mnemonics are stored in SUPERTAB rather than with the data set. This somewhat Rube Goldberg arrangement has, up until this writing, been necessitated by restriction of on-line storage within the WIT partition to source programs. The restriction against storage of object code within the WIT library has been lifted, and facility to store some data sets available through WIT is scheduled for the near future.

This is not a good general solution in that we cannot do it for very many surveys, but at the same time we should emphasize that there is no necessary reason for the WATFOR restriction on direct access data files. A batch system can handle such files perfectly well, and this would make it possible to achieve the same generality as in IMPRESS. The other side of this problem is that we are not persuaded that one needs a very large number of such surveys available for elementary analysis instruction, so it may well be that we shall achieve a quite satisfactory capacity for mnemonic variable identification with this arrangement.

With this means of variable reference, there will be very little difference between the two systems in the statement forms for specifying variables for tabulation. Actually, the IMPRESS format requires one more item of information, the direction in which percentages are to be run. Since the batch output for SUPERTAB is by lines, and the number of lines usually does not exceed the limits set, it is somewhat lavish and always gives percents both by rows and columns in each bivariate table, along with the observed frequencies in each cell. (This means that dependency relationships are permitted to be ambiguous.) Also, the marginals in each bivariate table are given both as frequencies and percentages. IMPRESS which has typewriter output by characters requires the one additional specification of percent direction and it only prints what is explicitly requested. One can find pedagogical reasons to do this in either way, but one suspects that each decision was made on the basis of what the hardware could and could not easily do. We shall return to printing differences below, but let us continue with our appraisal of the forms of input specifications.

The specification of statistical options is about the same in the two systems with regard to the intelligibility of the declarations that obtain them, the ease of making such declarations, and the presentation of the statistics in the output. SUPERTAB has one card, supplied by the user, which begins with the word "STATISTICS" followed by key words in any order and in free fields separated by blanks. The IMPRESS system uses similar mnemonic devices, but key words follow on the same line as the declarations of variables and task requests. The result of the specification arrangements for both systems is the placement of statistical output at the bottom of each table to which the statistic refers.

It is our opinion that the two systems are very nearly on a par as regards their demands on the user for specifications of task relevant requests, and much of their processing of such requests. Most importantly, many of the shortcomings of each arise from specific program design rather than from general system characteristics. Thus we may state that although instant batch and conversational systems are not identical, and they do not conduct their business in the same way, very much the same minimization of user costs and maximization of results can be achieved on either system for this kind of introductory instruction in data analysis.

There is another feature of these systems that we want to comment on that does not have to do directly with task specifications, but that does have a bearing on the use of results. It is that interactive systems that use a typewriter as a user console build in a feature that is very useful both for teaching and research. When the user is done with his session on the computer, he automatically comes away with a "hard copy" (typescript) of all transactions with the computer. Though batch systems often fail to give this kind of record, it is not intrinsic to their makeup. In SUPERTAB, all of the cards that are prepared by the student are listed on the output as they appeared in the input. This does provide an analog to the interactive copy, and it is quite essential to quick analysis of student errors for corrective instruction.

Costs

An important point of comparison of WIT and IMPRESS that we would like to be able to discuss is the one that is too often deisive, costs. Unfortunately, there are a number of complicating factors in attempting to make any firm assessments here, but we shall at least share our guesses, though they may not be very good ones. There is no difficulty in getting quite accurate statements of charges for particular jobs on each system, but it is very difficult to decide how to compare the jobs as such since they do different amounts of different kinds of work, and charges are not broken down by the categories of tasks that are directly comparable in the two systems. In an illustration of IMPRESS performance (Cline and Meyers, 1970), it appears that two three-variable elaborations (or at least most of the tables that would go into them in SUPERTAB) cost about $4.25. (Meyers, 1970b, pp. 8.27-8.29) Our best estimate is that the charges under WIT would have been about $3.00 (again the tables would contain more information).

Dollar costs ought not to be the only cost factor which serves as a point of comparison between the two systems. The student's time is a scarce commodity. The value of a computer system in instruction may be judged in terms of the amount of time added to, or taken away from, concentration upon the substantive issues of a course. Of course, we do assume that use of computing facilities reduces the amount of time that might be spent doing analyses or statistical computations with desk calculators or unit-record equipment. Yet use of computing equipment does not eliminate such "wasted" time altogether, and the type of system which commands less of the student's time is a superior tool. Again, as in our discussion of the dollar costs, a true comparative analysis of the two systems with regard to time costs cannot be performed because there are differences in the kinds of things the user and the systems do. In order to produce a three-variable "elaboration" with all tables, percentages, and statistics using SUPERTAB takes about one minute of the student's time at the terminal, and the same analysis using IMPRESS would tke considerably longer at the console. But if one takes into account that the SUPERTAB user must spend time punching cards (usually between two and six) the difference between the two systems decreases. Of course, over all, the important point has already been anticipated--both kinds of systems are an order of magnitude faster than any alternative.

Other Differences

While SUPERTAB seems to be somewhat cheaper and we cannot presently establish a definite difference in user time required, it is true that IMPRESS is capable of handling larger samples of data points in terms of the present implementation of these two systems. (See again note 7.) Usual criteria for evaluation of systems are that larger values of parameters are preferred. But another side to this question has been raised elsewhere (Sim, et al., 1970), viz., how many observations do we need to make available to students who are learning data analysis? Is "more" automatically "better" for these purposes? Or is there an optimal balance between having a sample that is large enough to give something to work with, (i.e., that will not be subject to such wide confidence limits as to be useless) but no larger, so as to minimize its costs in use? We are already on record above in favor of the latter position. In the logical extreme, we doubt that anyone will seriously suggest letting every college sophomore work on line with the one-in-a-thousand census sample; there has to be some sample size which is adequate for any given instructional purpose, and usually this does not have to be very large, since it is not necessary to use as many variables for exercises as one might want for a fullblown explanation. Teachers of research methods will probably disagree about optimal numbers, but for most of our own purposes we are satisfied with the present capacity of WIT.

IMPRESS can also produce cross-tabulations of more than three variables at a time, while the present version of SUPERTAB does not do so. However, this is not a necessary restriction, and the program is being rewritten to provide for higher order tabulations. This is being done principally in response to the needs of users whose interests are in advanced instruction and research, but also for the elementary instructional needs we have emphasized here. Another related characteristic of SUPERTAB is that only one multivariate tabulation will be performed in an individual run. Habitual users of conventional batch systems seem inclined to view this as a severe restriction, and it can be changed. However, since WIT is self-service and response is immediate, additional analyses can be had easily and quickly. Also, we feel that it is consistent with the objectives of training in elementary data analysis to emphasize serial examination of relationships and hypotheses.

Concluding Remarks

In order to summarize our comparison of IMPRESS and SUPERTAB in context, we must go back to the question of the value of batch processing as a pedagogical tool. It was noted that common opinion holds that non-interactive (or non-conversational) computing systems are less than optimal for instructional purposes. The first point upon which this argument is based is that batch systems are not easy to use, especially from the point of view of the inexperienced student. As has been pointed out above, we feel that SUPERTAB is as easy to use as IMPRESS. SUPERTAB requires no knowledge of the technical features of the computing system, nor does it expect familiarity with jargon associated with programming.

The second assumption is that turnaround must be fast because, for instructional purposes, immediate response is more effective as a corrective and reinforcement than is delayed response. Turnaround on the WIT system is immediate.

The final assumption is the most crucial one, especially from the viewpoint of the small college or university. Roughly stated, the two main points are: (1) conversational systems are exceptionally useful as aids to social science instruction, but they are relatively expensive; and (2) batch processing systems, because they lack such features as mentioned above, are not particularly well suited as pedagogical tools, but they are relatively inexpensive. But WIT offers the speed and relative ease of use usually associated only with conversational systems, and the cost associated with batch processing systems. Consequently, we feel that a system like WIT (with such programs as SUPERTAB) is as useful a pedagogical tool as is IMPRESS for instruction in social science data analysis. This taken together with the probable lower cost of instant turnaround leads us to believe that consideration of it as an alternative to conversational computing (in either case probably on a network basis) is warranted for the cost-conscious small college or university.

FOOTNOTES

1. In fact it probably gives faster response to the job once it is completed and submitted to the machine than does the usual time-sharing console that consists of a typewriter, since output is by lines in WIT, rather than by characters. However, volume output on a conversational system could be directed to a line printer, so this is not a really substantial difference.

2. This question is elaborated in Sim, Williams, and Kreider, 1970.

3. A feature that we have added very recently is a separate program that reads the parameters, but does not execute the job, and rather gives extensive diagnostic analysis of any errors found in the parameters. This will be used both for teaching and for users to debug their own mistakes when they presumably "know" the program.

4. We share the guilt for this with colleagues elsewhere; we have not tried to make a genuinely objective study of the effects of WIT--our whole discussion here is based much more on anecdotal material than it ought to be. On instant turnaround see Irons, 1965; Lynch, 1966; Smith, 1967.

5. We shall not attempt to reproduce here the documents and examples of input and output on which the ensuing discussion is dependent, since they are rather bulky. We would be pleased to provide copies of our material to readers who are interested in these questions.

6. The reader may discover this for himself by trying to decipher the IMPRESS examples in the appendices of Cline and Meyers, 1970, and Meyers, 1970b, without recourse to the textual explanations--or even with them. And we should emphasize that we think that IMPRESS is a relatively clear and straightforward system; there is an amount of instruction that must be done with any system.

7. However, there is one notable difference between the possible files that must be accounted, and it has to do with the maximum number of observations that are permitted in each of the cases. Because of differences in the way the two programs process the files, they differ as to constraints on numbers of data points and numbers of variables, IMPRESS requires that the user select a file of from two to eight variables from all of those available in the parent survey before any processing is done. This is stored as a contingency table of not more

than 4,000 cells in the computer's memory and saved for the user on a direct access device. (Note that these are cells of a master table, not respondents or datums.) For practical instructional purposes, there is no noticeable limit on the number of data points that may be included, because data points as such are not saved. The price is the limit of eight variables. On the other hand, in SUPERTAB, the user may call at any time for any of the 40 variables that are in the sample, but data sets must be limited as to the number of observations. An instructional system that gives quick response at reasonable cost cannot allow unlimited numbers of data points and variables. Which should be given up is partly a matter of objectives and philosophy, and partly a function of technical characteristics. In our present system it would be difficult to implement the IMPRESS solution; we suspect that handling large numbers of variables might be difficult in the Dartmouth time-sharing environment. Ideally, one ought to have the option of doing whichever was appropriate within the same system.

8. In SUPERTAB the codebook is not resident on line. We probably ought to comment on this since research oriented interactive systems (e.g., ADMINS, OSIRIS) do provide for machine readable codebooks. It is our view that there is a difference between research and elementary instruction, and the processing capacities that each entails. In general, one assumes that the researcher can pay for whatever will expedite his work, or at least that he ought to have the powerful options at his command if he can pay for them, and powerful options for non-numeric tasks do tend to be expensive. Since elementary instruction mainly uses reduced files, and since it concentrates on analysis rather than variable transgenerations, and since costs for large numbers of student users must be attended closely, there is little justification for providing the more sophisticated codebook handling abilities. But this is an opinion and it must be emphasized that instant turnaround batch systems that employ one-user-at-a-time terminals cannot provide this capability gracefully. Conversational systems seem to have a clear advantage here.

9. We have not compared data recoding statement formats. The procedures of each system are discussed elsewhere. (Sim, et al., 1971.) Here it is sufficient to say that the procedures are somewhat different, they are more elegant in IMPRESS, but both are adequate for elementary instruction in data analysis.

REFERENCES

Cline, Hugh F. and Edmund D. Meyers, Jr.
1970 "Problem-solving computer systems for instruction in sociology." American Sociologist 5 (November): 365-370

Gold, Michael M.
1969 "Time-sharing and batch-processing: An experimental comparison of their values in a problem-solving situation." Comm. ACM 12, 5 (May): 249-259

Irons, E. T.
1965 "A rapid turnaround multi-programming system." Comm. ACM 8, 3 (March): 152-157

Kreider, Glen D., Francis M. Sim, and Anthony V. Williams
1970 "Instant turnaround in instructional computing: some examples from the social sciences at The Pennsylvania State University," pp. 8.16-8.22 in Proceedings of a Conference on Computers in the Undergraduate Curricula. Iowa City, Iowa: The University of Iowa.

Lawrence, Sara
1970 "The Computer and the Liberal Arts: A Compatible Partnership." Carnegie Quarterly 18, 1-2 (Winter-Spring): 6-7

Lynch, W. C.
1966 "Description of a high capacity fast turnaround university computing center." Comm. ACM 9, 2 (February): 117-123

Meyers, Edmund D., Jr.
1969a "Project IMPRESS: time-sharing in the social sciences," AFIPS Conference Proceedings 34 (May): 673-680.
1969b "Survey of Social Science Computing Systems." Project IMPRESS, Dartmouth College, Hanover, New Hampshire.
1970a "Interactive systems and social science research and instruction." Social Science Information 9, 3 (June): 157-171.
1970b "IMPRESS and undergraduate education in the social sciences." 8.23-8.29 in Proceedings of a Conference on Computers in the Undergraduate Curricula. Iowa City, Iowa: The University of Iowa.

Rosenberg, Morris
1968 The Logic of Survey Analysis. New York: Basic Books.

Schatzoff, M. R. Tsao, and R. Wilg
1967 "An experimental comparison of time-sharing and batch-processing." Communications of the ACM 10, 5 (May): 261-263.

Sim, Francis M., Anthony V. Williams and Glen D. Kreider
1970 "Instant batch: halfway house or a third choice?" Mimeo.

Sim, Francis M., Gordon F. DeJong, Glen D. Kreider, and David E. Kauffman
1971 "Data analysis for sociology undergraduates." American Sociologist in press.

Smith, Lyle B.
1967 "A comparison of batch processing and instant turnaround." Comm. ACM 10, 8 (August): 495-500.

Williams, Anthony V., Glen D. Kreider, and Francis M. Sim
1970 "WIT at Penn State: a survey of the WATFOR instant turnaround system at The Pennsylvania State University." 11.1-11.8 in Proceedings of a Conference on Computers in the Undergraduate Curricula. Iowa City, Iowa: The University of Iowa.

COMPUTERS AND UNDERGRADUATE TRAINING IN AN UNDERDEVELOPED AREA: THE CASE OF LOUISIANA STATE UNIVERSITY IN NEW ORLEANS

John K. Wildgen
Department of Political Science
Louisiana State University in New Orleans

If the behavioral revolution's impact upon political science can be compared to the impact of industrialization upon society then the increasing use of computers in undergraduate curricula certainly signals the beginning of an "affluent society" for political science. Nonetheless, the process of modernization within our discipline has brought with it discontinuities and inequalities analogous to those existing within modernizing societies. At the current stage of development in the use of computers in undergraduate social science curricula it is only proper that emphasis be put upon the exchange of information regarding the employment of the newest, most powerful, and most sophisticated innovations in computing hardware and software. But some attention must be paid to those sectors of academia not yet blessed with access to facilities similar to those existing at Ann Arbor, Palo Alto, Durham, Hanover, or elsewhere.

There remain many competent and well trained political scientists and sociologists at institutions which for a variety of causes still have minimal computing facilities. To these men and women, users' manuals for OSIRIS, TSAR, SPSS, IMPRESS, and so on are like a Sears catalog in Appalachia--they provide merely a vision of a distant world whose benefits are available mostly through charity. Many of these people have become self-taught systems analysts, and through skillful exploitation of the facilities available to them they have been able to achieve research goals which would involve merely routine computational burdens at larger installations. Yet it is only understandable that a social scientist can tolerate much more adversity in his research than in his teaching. Normally, research tasks are not straight-jacketed by the inevitable time and schedule limitations of the semester or the quarter. For research purposes it is a small sacrifice to await the wee hours of the morning in order to find enough core or peripheral devices for a particular analysis. But this mode of operation cannot be reasonably expected of undergraduates.

Aggravating the problems caused by the lack of physical computer size--problems in the order "capital"--there exist at many American campuses problems of a more human nature. In the middle and late 60s the computer began to assume in the popular mind the role which had previously been played by the bomb. Dr. Strangelove was dethroned by HAL and many students began sporting "Do not fold, spindle, or mutilate! I am a human being!" buttons. Automated registration procedures at some of the nation's finest universities inspired student leaders to denounce the computer with all the fervor of William Jennings Bryan attacking the gold standard.

The use of computers in undergraduate instruction implies a widening of the technological and methodological base of the social sciences. Considering the vital research applications of computers, their use by students implies an unprecedented participatory role for students in research techniques, such as causal modeling and latent structure analysis, which were considered rather exotic as late as the mid 60s. Thus we can pose the question: If "a participatory political system requires as well a political culture consistent with it," could it not be true that a participatory computer system requires as well a computer culture consistent with it? A computer culture would seem to be paraphrasing Almond and Verba, "a humane and conservative way to handle quantification and data processing." The development of a civic culture appears to be contingent upon a society's ability to limit the trauma associated with the conflict of traditional institutions against the dynamics of modernization. Likewise the computer culture would seem to develop best in areas where the conflict between art, manners, and agriculture on the one hand, and engineering, morals, and industry on the other hand was not traumatic.

With these two problems serving as background we can present the fundamental questions to which this essay is addressed: how can computers be used in undergraduate training within a context of severe capital and cultural deprivations, and how successful is it when attempted? Our procedure will be straightforward and will involve a brief account of how the capital limitations in question came about and what they are, how the absence of a computer culture has manifested itself, and how we have moved to confront these problems. We will finish with a brief and, hopefully, fair assessment of our successes, failures, and hopes for the future.

The "universe" in this case is Louisiana State University in New Orleans, an eleven year old urban campus which, like many campuses similarly situated, has encountered unexpected growth since its inception. In most respects it is not at all atypical of other American campuses aside from the fact that it is in Louisiana, a state still deserving of V. O. Key's apt label, "The Seamy Side of Democracy." While this is not the proper forum for a discussion of the vicissitudes of Louisiana politics, it still must be pointed out that state politics provide a major element in the state's educational environment. As with the question of wealth in Louisiana, the computational problem is not so much one of lack as it is of distribution. Among the many institutions of higher learning in the state, a few possess a capital investment of computing machinery large enough to make those campuses the envy of many even in the North. Yet pleas for a more even distribution of computer power have met with a response best described by the cursive prose of a common System 360 error message, "input data stream flush."

Perhaps one can gain a more concrete idea of the situation we face by trying to comprehend our reaction to an article describing a computer system in use at Dartmouth College. The major part of the article was devoted to a student's hypothetical use of a very convenient cross-tabulation routine. In and of itself, the obvious use of sophisticated programming techniques was quite impressive. But what caught our eye was the appended log-in and file selection section. We noted to our amazement that the system statistics indicated fifty-two users on line at that time. Our system is limited to a maximum of sixteen, of which only five can enter from remote terminals. We noted to our further amazement that the data file in use was the 1968 Michigan SRC election survey stored, presumably, on disk. At our installation a single box of punchcards occupies more than 10% of the available disk. This makes it impossible to allocate any permanent storage or to comtemplate even temporary storage of an entire election study. Indeed, since we have no magnetic tape capability, complete election studies of the SRC's 1968 election study's scale cannot be processed at all. (1)

Leaving the question of physical capabilities and turning to the attitudes of the students we find a similarly distressing problem. There is no evidence that Louisiana's college students possess any more or less native intelligence than students elsewhere. But when it comes to computers they are obviously culturally deprived. Despite a keen (indeed remarkable) interest in political behavior, stemming from a regional political culture which decidedly could not be called "civic" in Almond-Verba terms, the idea of using computers in political research truly frightens them. Due to the fact that improvements in LSUNO's computer installation have allowed us to update courses more rapidly than their descriptions in the university catalog, a considerable number of students have registered in our computer-oriented courses in political behavior while ignorant of the fact that a computer will be used in the classroom. It has been our policy to make clear at the first class meeting that a computer will be used, that it will be a labor saving device, and that no prior knowledge of computers is assumed. Despite this softsell approach we have seen initial registrations approaching thirty students contract to eight within two days. Due to the fact that rumors about courses travel much slower at commuter campuses than at residence campuses, the above described phenomenon has not entirely disappeared. Consequently, we have built up a backlog of computerphobil students. Extensive informal probing invariably leads us to the same conclusion: two-thirds to three-fourths of our students are simply scared stiff of computers. No amount of cajolery will suffice to calm their fears.

In a sense very little has been done beyond merely probing student fears of the computer. We recognize these fears as a future limiting factor in curriculum development, but for the time being we have found it convenient to allow these students to remain unmobilized. The last thing we can afford is an explosion of demands for computer-oriented courses.

For the students who remain interested in integrating the computer with the other analytical tools used in the social sciences we have designed two courses which take into account our limited computer power and their lack of familiarity with computers. It bears pointing out that even those students who decide to take a course which uses computers are hardly universally enthusiastic over the prospect. Some are, of course, but many of them remain quite wary. Given this atmosphere it is essential that introductory lectures be quite well planned and that the instructor be intimately familiar with each student's progress and state of mind. We have found that no commitment can be made to any kind of time schedule in presenting introductory material. In our experience, some classes have been much slower than others, for reasons still unclear to us. What is clear is the non-utility of moving too fast.

Our first course, entitled simply "Political Behavior" is composed of three phases. The first is a review of what the students already have learned of political behavior in lower level courses. The purpose of this phase is not to turn the student into a walking bibliography, but to make him curious as to how certain propositions about political behavior arose. Use is made of brief texts such as those by Bowen (2) and Flanigan (3).

The students are deliberately led into discussions of the material covered in the texts and it is at this point that the students can be indirectly steered into very fruitful encounters with the problem of operationalizing research on a specific hypothesis or set of hypotheses. In sum, they begin to acquire an intuitive feel for the why of behavioral research, especially through survey research.

When this point has been reached, and only at this point, the students are introduced to the data file they will be working with and the programs they will employ. It is regrettable that we are restricted to using only one source of data at this level, but it is an inescapable consequence of our limited installation. Its shortcomings have already been hinted at, but some more elaboration may prove helpful. LSUNO's only research computer at all suitable for classroom use is a PDP/10. Perhaps the best way of describing the machine to those unfamiliar with it is to compare its cost with the powerful IBM 360-65. LSUNO has invested roughly $500,000 in its PDP/10, while LSU-Baton Rouge has invested roughly $3,500,000 in its 360-65. The PDP/10 is a useful machine, but LSUNO's plant is clearly a minor one.

The data file we employ is the one which accompanies Flanigan and Repass' innovative text, Electoral Behavior. (4) This file has several important advantages in the classroom use of small-scale computers. First, it is brief--occupying no more that 12% of the system's available disk space. Secondly it includes most of the theoretically interesting variables affecting voting behavior which could be successfully used by undergraduates in an introductory course of this nature. Third, it combines the results of surveys for several election years, enabling students to compare the impact of variables across time. Fourth, the data is all single-column, greatly simplifying the inevitable problems students confront in mastering format statements. Most importantly, it stimulates the student's curiosity and generates demands for more data.

The first step in orienting the students to the use of survey data is teaching them how to use the marginal frequencies reported in the codebook included in Flanigan and Repass' text. The meaning and the research value of marginals is virtually second nature to behavioral scientists familiar with survey research, and there is a great temptation to charge ahead and start working with contingency tables. But we have found it worthwhile to take extra time to familiarize the students with the marginals. It gives them something of a feel for the data, helps them to develop hypotheses which the data can help them test, and (incidentally) gives them a chance to practice calculating percentages by hand--frequently a lost art.

Once the students feel comfortable with treating marginal frequencies they are introduced to some sample output from the cross-tabulation program they will eventually employ on their own. This allows us to confront two inevitable problems arising among students heretofore unfamiliar with this type of research: reading computer printout and drawing their own inferences from a contingency table. The vast majority of our students have never seen computer printout other than in their grade reports. For some, fortunately a few, it is a disorienting experience.

In designing a suitable format for printout of contingency tables we had to engage in a form of intuitive linear programming, given the number of constraints we operate under. The program had to be designed while keeping in mind the limited user area available on the PDP/10, some peculiarities in the Flanigan-Repass data set, and a number of pedagogical considerations. Because of limited machine size and the lack of peripheral equipment it was not possible to design a program which created several tables simultaneously in core; rather we had to read in data, create a matrix, write output, and then rewind for each table desired. This strategy saves memory but is painfully I/O bound. Moreover, since disk space is limited it was impossible to even take advantage of binary rewrite capabilities in the compiler, thus forcing us to use the least efficient tactics in an inefficient strategy.

The data presented some problems of its own. Since the Flanigan-Repass data combines four surveys, each with different "Ns", it is unavoidable that the data cards will be padded with blanks in those fields not allocated to the surveys with larger numbers of cases. Since many data storage modes do not distinguish between the character '0' (zero) and the character ' ' (blank) we ran the risk of confusing legitimate zero codes with blank padding. One alternative considered was simply splitting the data into four smaller files, but this would have forced the students to modify their codebooks in order to keep column locations straight. This was deemed a possible source of confusion so it was decided to keep the file integral. That decision committed us to the use of alphanumeric variables and formatting, once again a method costly in terms of efficiency.

The above constraints, however, did not prevent us from designing an output format with several advantages from the students' point of view. In the first place the table uses the same codes for column and row headings as are found in the data. That is, headings are not ordinal (as is the case with a program like NUCROS) but are picked up from the data. Thus if a researcher codes a particular variable 1, 3, 5, 9 then the table will use the same characters and not convert them to a 1, 2, 3, 4, 5 ordered series. This saves students from much confusion in relating the results of the cross-tabulation to the original marginal frequencies.

Some other major considerations which went into our design of the output format can be gathered from a quick glance at the facsimile output in Figure 1. In an early version of this still experimental program both row and column totals were provided. Subsequently the row totals were deleted to encourage students to get into an early habit of reporting percentages calculated columnwise. The program itself does not present percentages, merely the frequencies. At this initial stage of working with data it has been found that several advantages follow from this approach. By working with the frequencies the students quickly learn to identify modal cells, without having to weight percentages by marginal totals. The modal cells are rendered immediately apparent. This approach pays off later on as well when discussions of significance arise--discussions which can only be complicated if the students are not used to working with frequencies.

Figure 1

Sample Output

```
ROWS ARE VARIABLE
RELIGION IN 1964

COLUMNS ARE VARIABLE
PARTY IDENTIFICATION IN 1964


ROWS = ITEM 5        COLUMNS = ITEM 6

CODE           0    1    2    3    4    5    6    7    8    9

         504   0    0    0    0    0    0    0    0    0    0

  0        0   0    2    0    0    0    1    1    0    0    0

  1        0   0  102   93   36   27   12   26   28    0    2

  2        0   0    0    2    1    1    2    1    1    0    0

  3        0   0   10   10    8    6    1    3    0    0    0

  4        0   0   10   12    3    5    4    4    6    2    1

  5        0   0   34   58   20   18   25   57   57    1    0

  6        0  10  187  148   39   27   26   77   53    2    3

  7        0   2   35   26    4    7    5   14    5    0    1

  8        0   0    5    7    5    4    0    7    5    0    1

  9        0   1    9    8   12   10    5    5    2    0    0

N =      504  13  394  364  128  105   81  195  157    5    8
```

Another aspect of this program is that no cell collapsing options are made available at this stage to the students. This forces the students to consider carefully the problems of operationalization involved in collapsing for purposes of clarity, and the pitfalls involved in drawing inferences from poorly thought-out attempts to collapse.

Our procedure is to provide the students with several (usually four) initial tables and give them a weekend to prepare finished tables and a short report on what information the tables contain. Fortunately the small class size allows the students to present their findings in turn. The students themselves become their best critics and usually one can count on at least two entertaining class periods as the students catch each other's mistakes and debate alternative ways of presenting data and interpreting results. By allowing students this freedom of participation one also avoids risking a possible conceptual divorce in their minds between methodology and political analysis. From the beginning they grasp the importance of subordinating technique to substantive goals. This has direct benefits later on when the students begin to consider a term project. They are aware that there is the possibility that inferences about political behavior may be based in some cases on artifacts of method. Thus they develop a healthy, critical attitude towards the literature, and the need for replication and reanalysis becomes apparent to them.

At this stage the students' attitude towards the idea of initiating their own term projects is one of general enthusiasm which makes the next stage in instruction, confronting them with the computer and programming techniques, much easier. In developing our program considerable effort was expended on making the program control process as logical as possible. One of the most fortunate aspects of the PDP/10 is its conversational, interactive, mode of operation, making it possible to design a program which leads the student through a series of questions to which he types the appropriate answers. Of course it is better if the student already has the answers written down--at a minimum it reduces time spent sitting in front of the teletype. With this end in mind we designed a job form keyed to the control sequence of the program. Normally a single class period is more than sufficient for explaining the form and the students are asked to design a test run. The program is currently dimensioned to handle as many as 50 variables though, frankly, no one has ever taken advantage of this capacity. It is quite flexible in that any combination of tables can be made and the number of control variables that can be used in an analysis is limited only by considerations of practicality. However, when controls are introduced the student must execute the program for each range of values in the control variable(s) he considers significant. Thus if a student wanted to analyze the relationship between occupation and party identification controlling for region he would have to decide how to operationalize region and then execute separate analyses for each regional condition he wished to use. The control sequence has been designed to allow collapsing so the student could dichotomize the regional variable into "North" and "South" by setting range signals (lower limit and upper limit). The control routine used has the unfortunate property of exercising the control over every table called for in the analysis, however. Essentially the control routine simply excludes data from the entire analysis. If the value of the control variable read on a particular record does fall within the limits set the machine simply skips the record and goes on to the next, making it impossible to include both controlled and uncontrolled tables in the same run. This disadvantage is not severe in view of the overall success of this program with the students. Indeed, one major advantage of this property is that it forces the student to think quite deliberately about why he is using a control variable, and the problem of spuriousness.

In general our experience with this program has been quite good with beginning students. Its simplicity of operation coupled with the combination of advantages and limitations, has forced the students to confront many of the major problems in the execution of survey data analysis while not intimidating them with the complexities of more sophisticated programs--nor lulling them into a false sense of competence by providing them with the luxuries of pre-operationalized collapsing.

Nonetheless, we are continually experimenting with this initial program as we discover unforeseen pitfalls--and unanticipated aptitude in our undergraduates. This experimentation usually takes place during the period of the course the students devote to performing their own analyses. Many class sessions are spent actually interacting with the computer. While some students correct their inevitable errors in programming and others work with the terminal, there is generally time for the instructor to devote individual attention to students having serious problems at a particular phase in their project, or to talk over with some of the students an interesting finding by one of their colleagues. This is a period of intensive learning for them and certainly the most satisfactory part of the course. During this period the concept of statistical significance invariably arises. It has been our practice so far to shy away from elaborate discussions of chi-square tests and orient the students toward tests of association such as tau and gamma. Some of the students are, understandably, much more sophisticated in their knowledge of statistics than others and they prove to be quite an asset to their peers. For those who are interested in a more statistical approach to an interpretation of the data we have employed a number of useful library routines but we have not yet found it necessary in this first course to do more than broach the topic.

In assessing the overall success of this course we have tried to keep in mind two considerations: what has been done for the students' level of knowledge, and how much has their interest in quantitative techniques been stimulated. Frankly, we really have little to go on in answering the first question aside from a claim to the dubious title of being first and foremost in Louisiana. But as for their interest in more advanced courses, we can report unambiguous success. Because of this we have introduced computer usage into a second course entitled Comparative Political Systems.

The strategy of the second course is based upon the assumption that the students no longer fear the computer and are, indeed, eager to exploit its potential as a tool. Thus we have introduced a wider variety of data sets and techniques. In the area of voting behavior and public opinion we are making use of a series of Gallup Poll data and custom analysis decks from the ICPR. We have also adapted the NUCROS contingency table generator for our use. As an aid to both students and faculty we have developed a conversational program which creates a NUCROS control card deck which can be saved on DECTAPE or punched tape and modified in part using a system text editor and corrector program. Generating control cards for a program like NUCROS can take some time at a console so minor modifications (such as introducing new control variables for a set of tables) are generally done without creating an entirely new control file.

Many of the students, however, have exhibited an interest in other types of methods. The students seem to break down into several sub-specialties--those interested in mass behavior and opinion, cross-national comparisons, and elite behavior (including roll call analysis). This broadening of interests can only partially be accounted for by what they are exposed to in classes; much of it stems from an awakened personal interest in scientific approaches to political questions. We had never anticipated, at this stage, devoting a class at the undergraduate level to techniques such as factor analysis, multiple regression, and least distance analysis. But now we have students who have sought out, on their own, literature on these methods and their employment, and who are eager to try their own hand. Naturally, some disasters have occurred--witness, for example, the student whe went to great pains to gather roll-call data on the Louisiana Legislature, only to find that the thirty-nine member state Senate revealed a structure composed of 13 factors which accounted for less than half the variance--something which told us nothing new. It was a difficult, but not vain, lesson in the limits of certain quantitative approaches. On the other hand, we have seen some notable successes, as in the case of the student who collected reams of Irish electoral returns and analyzed them using techniques developed by Douglas Rae. The results may well merit publication.

It seems fair to say that, in general, our efforts have met with success in that more good than harm seems apparent. The word seems to be spreading that while computers are inhuman, their users need not be. If this is the case the cultural flaw which nurtures an unjustified fear of computers may be corrected. Yet is is a constant source of frustration to be limited by an inadequately fleshed-out computer system--one with great potential but little accomplishment. On numerous occasions students have been frustrated by an inability to find a free line to the computer. When they do they are frequently paralyzed by insufficient disk space. We have considered the alternative of converting to a batch mode of operation (and we have done so in emergencies). However, that approach, after a taste of advantages of the immediacy of time-sharing systems, seems a horrible let-down. Moreover, due to the frequent mistakes made even by experienced computer users, batch processing what are essentially social-science experiments proves ultimately to be no genuine alternative. The 12 to 24 hour waits to uncover simple errors, which are immediately reported and correctible in a time-sharing mode, often idle the progress of a student who requires results as the next step in his project.

But not even a healthy dose of capital investment in our computer system, or indeed many others, would provide a full solution. A constant source of irritation to us is the fact that most of the genuinely sophisticated social science programming packages are machine specific. For example the OSIRIS package is currently restricted to users with ready access to an IBM 360 or some of the larger RCA machines. We well understand that there are legitimate economies to be had in resorting to assembly language programs or I/O routines in large-scale statistical packages. But this economy is bought at the cost of incompatibility with other computers. Technical solutions exist and in the long run they will be applied.

Still, we would like to make a plea at this point for several things which would enhance the entire picture for the use of computers in undergraduate training. First, more FORTRAN in large statistical packages. We do not advocate that installations using assembly language routines employ it themselves, only that they seek ways of making FORTRAN translations available to campuses with non-compatible machinery. Some institutions are already taking steps in this direction and we can only cheer them on.

A second thing which has occurred to us is the desirability of regional computing facilities serving several campuses at once. The economies of scale which can be attained here are obvious--as the political problems are too. The major obstacles are the natural conflicts arising from the different computing needs of various disciplines, some of which are compute-bound while others are I/O bound. Nevertheless, developments in computer technology promise to resolve the one time dichotomy between data processors and arithmetic machines.

A third recommendation we have is for continued development of conversational statistical packages such as IMPRESS. When reliable, capacious machinery is available the results which can be attained by the classroom use of timesharing computers are most impressive.

Naturally, in our position we are of the opinion that the measures we have suggested should be implemented with "all deliberate speed" in the non-Southern sense. Yet, in our desire to close on as optimistic a note as possible we would like to pass on the comment of a student of ours who has suffered from all the limitations of our system, "...best thing since the blackboard."

FOOTNOTES

1. Hugh F. Cline and Edmund D. Meyers, Jr., "Problem-Solving Computer Systems for Instruction in Sociology," The American Sociologist, Vol. 5 (November 1970), pp. 365-370

2. Don R. Bowen, Political Behavior of the American Public (Columbus: Merrill, 1968).

3. William H. Flanigan, Political Behavior of the American Electorate, (Boston: Allyn and Bacon, 1968).

4. William H. Flanigan and David Repass, Electoral Behavior, (Boston: Little Brown, 1968-69).

WE DON'T KNOW WHAT WE ARE DOING*

Edmund D. Meyers, Jr.
Dartmouth College
Hanover, New Hampshire

This paper is primarily a plea to my colleagues to gather good, solid information about our computer-related activities. That is, if we are lacking meaningful statistical evidence concerning the computer efforts of our undergraduates, then we do not know *as precisely as we should* what our students are doing. A fairly convincing argument can be made that, if we do not know what our students are doing, then we do not really know what we are doing either.

Following a discussion of the reasons for collecting such data, I will report on the successes and failures of a full-year effort to accumulate information about user activity. By way of a sneak preview of the results, it turns out that I do not know what I am doing either.

Background

At the June, 1970 edition of the Conference on Computers in the Undergraduate Curricula in Iowa City, papers were presented concerning a number of social science software systems. Specifically, there were: 1) the WATFOR Instant Turnaround (WIT) facility at Pennsylvania State University, 2) Project IMPRESS at Dartmouth College, 3) the APL programs at Cornell University, and 4) the LANSOR system at the University of Minnesota. Additional systems exist in the social sciences, from ADMINS to UTILITY CODER/360. These are elegant and useful packages. How do we know this to be true? We designed and implemented them; so they must be elegant, and they surely are useful.

What hard evidence is there about this work? We know that our undergraduates use our systems because they find a certain sadistic pleasure in uncovering programming bugs and showing these problems to us. We also find masses of output submitted with weekly projects and term papers. We observe students queued up at batch stations or pounding with a mite too much gusto on time-sharing terminals. There is all the evidence in the world that our elegant systems are being used by our eager, young intellectual disciples.

In a different vein, one extremely important indicator is one's subjective assessment of the results. If students are writing more provocative papers or if they appear to be learning more or possibly if our enrollments are increasing over time, then we might tentatively conclude that the machine is an asset to our pedagogy. Honest empirical measures of intellectual productivity have successfully eluded educational researchers thus far; so we will have to be content with our subjective evaluations.1 The collective appraisal of the papers at this conference is an extremely positive one. (1)

Let us grant both that there is some superficial evidence that undergraduates are using our software packages and that there is subjective evidence that our students are learning more. Do we know much about how many attempted runs were made prior to the attainment of the results submitted with the final paper? Who is using our software? How frequently is it being used? How efficiently? What do we know about the costs of these computer-related activities? Solid assessments of the dollar costs are rare. (2) Evaluations of the cost to the undergraduate student--in terms of frustration, alternate activities, inefficient operating systems, error messages which cannot be deciphered, etc.--are especially rare.

We are considering software systems which generate statistical analyses from social science data files. Isn't it at least curious that we are willing to include an amazing diversity of data files, *except* data about our own systems?

Privacy

I propose that the reason we do not gather information about the activities of users of social science software packages is primarily a mechanical one. It is difficult--especially when each user has his own account number--to gather the information in a single place; however, with some persistence, these technical problems are often surmountable. Privacy should be the salient issue.

Each undergraduate using a computer has the right to be witless or tedious in total privacy. This is one of the more important advantages of computer usage: The slower student may take longer to reach established goals, but he can still get there; furthermore, he can reach those goals without the embarrassment of appearing dull to his mentors or his peers. If we were to invade this privacy, we would quickly defeat our initial goals.

The more standard questions of privacy are also relevant here. Monitoring individual behavior, even if that behavior is as harmless as the execution of a "canned" program, is serious business of the "Big Brother" variety.

Thus, if we are to gather data about our users, it is imperative that we examine only their collective usage and that we totally avoid the monitoring of individuals *per se*. Although this approach does not totally divorce one from invasion-of-privacy questions, it does reduce that threat to a tolerable level. (3)

IMPRESS Usage

Thus far, it has been suggested that user information be gathered and that it be done in such a fashion that individual privacy not be invaded. This is one of the few instances in which I have followed my own advice; so we will now turn to the results.

Project IMPRESS at Dartmouth College was initiated in June of 1968. (4) The goal of this project is to provide social science data files and software for the analysis of relations among variables within those files; this is being carried out on the Dartmouth Time-Sharing System so that the facility may be widely used within undergraduate courses. The first year or so of the project was devoted to system design and implementation; by June of 1970, the basic structure of the software was stable, and the analytic capabilities have been gradually expanding ever since. Starting approximately June of 1970, minor modifications were made to the software such that entries would be made in a library data file each time the system was executed. A simple method of summarizing information about user activity was implemented, and we started accumulating data about IMPRESS usage.

Project IMPRESS supports three closely-related software packages. The first, IMPRESS***, is the major system. The second, XIMPRESS***, is an experimental version which is used to introduce and debug new analytic capabilities. Between academic terms, XIMPRESS*** is upgraded to IMPRESS***, and the cycle is started over again. (5) The third, EXPRESS***, is a *Reader's Digest* version of the main system, and it consists of a series of data exercises which have been partially pre-cooked by course instructors. As a result of the existence of three systems, usage statistics are recorded for each.

The first major step in data gathering consists of making entries in a library file whenever the software is executed. Actually, two entries are made in the library file, one early in the run and the second at the conclusion of the run; these entries are depicted in Figure 1. The entries include the following information. . .

First entry

the date (e.g., "01/18/71")

the time of day (e.g., "14:33:34" meaning 34 seconds after 2:33PM)

the individual's user number (e.g., "HDF33180")

the software system used (e.g., "I", "X", or "E")

the data file being analyzed

the number of variables included in the analysis

Second entry

the date

the time of day

the individual's user number

the software system used

execution time in C.P.U. seconds (e.g., "14.9622")

The technical ability to make these entries is provided by language BASIC as implemented in the Dartmouth Time-Sharing System (DTSS). (6)

Figure 1

```
OLD IMPRESS***
READY

RUN

IMPRESS   (COMPILED)  25 APR 71  19:14

PRIVACY WARNING

DO YOU WANT USER'S INSTRUCTIONS? NO

ENTER NAME OF STUDY? 20 HS

          HIGH SCHOOL STUDENTS--MCDILL-MEYERS-RIGSBY

ENTER NAMES OF 2 TO 8 VARIABLES TO BE RUN
? SEX,RACE,COLLEG,AR,MATH,POTI,BUTDAD              ********************
          <--------------------------------------* MAKE FIRST ENTRY *
ANALYSIS MODE                                      ********************
? DISC

ENTER MAPPING OPTION
? SG


ENTER COMMAND
?

     .
     .
     .

ENTER COMMAND
? STOP                                            *********************
          <-------------------------------------* MAKE SECOND ENTRY *
TIME: 20.562 SEC.                                 *********************
```

This scheme is rather similar to the one used by the telephone company with respect to direct-distance-dialing; a record is made when you start, and another record is made when you finish. By matching the entries, one obtains the desired information. Once each week, we execute a program which matches entries and produces a summary of activity for the 7-day period of Sunday through Saturday. You will note that the full user number must be recorded in order to do the matching; however, once the match is made, only the leading digit of the number (which indicates the type of user) is recorded, and then the original file is scratched. (7)

As a result of this scheme of recording information about usage activity, the unit of analysis is a completed run of IMPRESS***, XIMPRESS***, or EXPRESS***. That is, we are recording data about uses and not about users; while it is known how many times the system is run, it is not known how many individuals used it.

As another result of the recording scheme, the usage statistics are rather conservative, since it has not been possible to match all entries in the original file. For example, . . .

Report period: 6/28/70 - 4/17/71

Paired entries (start and completion)	8,934
First entry only (start--no completion)	10,193
Second entry only (completion--no start)	628

Unmatched first entries arise due to abnormal termination of program execution (use of CTRL-SHIFT-P or BREAK, exceeding run-time limit, encountering a fatal program bug, etc.). Unmatched second entries occur by starting the system in the middle and not at the beginning; this stunt requires considerable sophistication on the part of the user. Therefore, while it is not clear how much useful work may have been accomplished in those runs which appear as unmatched entries, it is clear that the remaining discussion of paired entries is a quite conservative estimate of total usage.

Given that data are accumulated on a weekly basis, let us begin with an examination of user activity from week to week. The relevant information is displayed in the form of a histogram in Figure 2. During the summer term, the usage was rather modest; in the summer of 1970, enrollments were modest, and course work tended to be both introductory in level and humanistic in content. During the Fall and Winter terms, it is rather clear when both mid-term and final papers and projects were due. Not surprisingly, usage of the system was quite seasonal.

Report period: 6/28/70 - 4/17/71

Minimum (week of 8/30/70)	4
Maximum (week of 11/29/70)	758

The IMPRESS software was designed primarily for undergraduate students, and the use of that software should vary with the academic calendar.

Instead of speculating about undergraduate use of the system, we can examine usage by the leading digit of the user number; this provides information about broad classes of users. It is observed that Dartmouth undergraduates account for almost three-fourths of that activity.

Report period: 6/28/70 - 4/17/71

Dartmouth undergraduates and summer term	71.0%
All other users	29.0%
Total uses for period	(8,934)

A detailed distribution of uses by leading digit of user number is shown in Figure 3.

Figure 2

```
(USES)    WEEK OF
                    I-----------------------------------------------------
  (77)   28 JUN 70  I*****
  (53)   05 JUL 70  I****
  (60)   12 JUL 70  I****
  (70)   19 JUL 70  I*****
   (9)   26 JUL 70  I*
  (47)   02 AUG 70  I***
  (80)   09 AUG 70  I*****
  (25)   16 AUG 70  I**
   (8)   23 AUG 70  I*
   (4)   30 AUG 70  I
  (13)   06 SEP 70  I*
   (8)   13 SEP 70  I*
  (18)   20 SEP 70  I*
  (97)   27 SEP 70  I******
 (231)   04 OCT 70  I***************
 (220)   11 OCT 70  I***************
 (511)   18 OCT 70  I**********************************
 (263)   25 OCT 70  I******************
 (304)   01 NOV 70  I********************
 (375)   08 NOV 70  I*************************
 (247)   15 NOV 70  I****************
 (417)   22 NOV 70  I****************************
 (758)   29 NOV 70  I***************************************************
 (477)   06 DEC 70  I********************************
 (215)   13 DEC 70  I**************
   (6)   20 DEC 70  I
  (47)   27 DEC 70  I***
 (146)   03 JAN 71  I**********
 (541)   10 JAN 71  I************************************
 (252)   17 JAN 71  I*****************
 (504)   24 JAN 71  I**********************************
 (301)   31 JAN 71  I********************
 (154)   07 FEB 71  I**********
 (314)   14 FEB 71  I*********************
 (381)   21 FEB 71  I*************************
 (543)   28 FEB 71  I************************************
 (611)   07 MAR 71  I*****************************************
 (136)   14 MAR 71  I*********
  (97)   21 MAR 71  I******
  (44)   28 MAR 71  I***
  (69)   04 APR 71  I*****
 (201)   11 APR 71  I*************
   (0)   18 APR 71  I
   (0)   25 APR 71  I
   (0)   02 MAY 71  I
   (0)   09 MAY 71  I
   (0)   16 MAY 71  I
   (0)   23 MAY 71  I
   (0)   30 MAY 71  I
   (0)   06 JUN 71  I
   (0)   13 JUN 71  I
   (0)   20 JUN 71  I
                    I-----------------------------------------------------

EACH  *  REPRESENTS 15 OBSERVATIONS.
```

Figure 3

System Activity by Leading Digit of User Number

(Report Period: 6/28/70 - 4/17/71)

Leading Digit	User Group	Uses	%
numeric	Dartmouth Undergraduate Students	6,124	68.5
A	Administrative Officers, etc.	9	0.1
B	Commercial Users	1	*
C	Kiewit Center - Software Development	278	3.1
D	Departmental & Graduate Students	266	3.0
E	Colleges on Dartmouth Network	791	8.9
F	Faculty	458	5.2
K	Kiewit Center - Admin. & Applications	30	0.3
N	NERCOMP Network	123	1.3
P	Research Projects	3	*
R	Research Projects	622	7.0
S	Summer Session Students	220	2.5
W	Faculty Wives, etc.	9	0.1
TOTAL for IMPRESS***, XIMPRESS***, & EXPRESS***		8,934	100.00

* Less than .05%

Now, let us examine which data files were used. This is of particular interest to the project staff, since the acquisition and preparation of data files is time-consuming and expensive. The most popular data file is the 1968 Presidential Election Survey which was conducted by the Survey Research Center at the University of Michigan. The popularity of this file seems to a function of 1) the availability of a codebook, 2) the recency of the data, 3) the excellent coverage within this survey of numerous bread-and-butter social science variables, 4) the salience of attitudes toward Viet Nam and related issues, and 5) the relevance of political data to the current student cohort.

In mid-December of 1970, when these data were examined for the Fall Term only, a relationship between usage of data files and the existence of codebooks was observed.

Report period: 9/27/70 - 12/19/70

12 data files with codebooks	84.3%
13 data files without codebooks	15.7%
Total uses for period	(4,088)

Of the 4,088 runs of IMPRESS*** and XIMPRESS*** during the Fall Term, 84.3% utilized data files where the codebook existed for the majority of the term. This finding was mildly surprising, since much of the codebook information is available through a time-sharing terminal as soon as a data file is placed on-line.

This data-gathering scheme also permitted us to consider the characteristics of the average run.

Report period: 6/28/70 - 4/17/71

Mean terminal time	14	min., 13 sec.
Mean C.P.U. time	15.3	sec.
Mean number of variables included	5.2	variables
Total uses for period	(8,934)	

On the average, the users spent less than fifteen minutes working at a terminal; they used about 16 seconds of computer time; and they included 5.2 variables in their analyses. Thus, at the rates "charges" students (no student ever sees a computer bill), the average run cost $2.16; however, this figure does not include disk storage, terminal rental, or line charges (all of which are monthly charges). In terms of what is being accomplished, I feel that this is inexpensive; the reader will, of course, reach his own conclusion.

Now, let us consider the same sort of information but for all uses of the system over the entire year. The total cost of providing IMPRESS service is dependent upon which of the three rates charged by Kiewit Computation Center is appropriate. Data are summarized in Figure 5. The "inside" and "outside" rates are intended for non-profit usage and are set such that the computer center will break even; the "commercial" rate is an approximation to what one might have to pay from a commercial time-sharing vendor. If these estimates are rounded, one can conclude that the level of usage of IMPRESS which we have experienced over a 12-month period would cost approximately $94,000 on one's own computer and roughly $154,000 from a commercial vendor. By itself, this seems expensive. Relative to what our colleagues in laboratory sciences spend per year to equip and maintain those laboratories, the cost of IMPRESS (which does service all social scientists at Dartmouth) is rather reasonable. (8)

Before summarizing the information just discussed, I would like to introduce some other data. Within the Department of Sociology at Dartmouth, many of us require students to complete course evaluations before grades are recorded. Individual responses are anonymous, and most of their comments are so candid and strike so close to home that most of us do not read the evaluations until we have submitted the letter grades to the Registrar. (9) Figure 6 displays the relevant results, summarized over a number of courses. Our students seem to find the use of IMPRESS within a course to be one of the more attractive features of the course. While this is gratifying, the data are inadequate to what we need to know.

Between the computerized monitoring of IMPRESS usage and the course evaluations in my department, it might appear that we know a great deal about the use being made of the IMPRESS system. In order to rapidly dispel that notion, let us consider some of the things which are not known.

First, we do not know how many individuals use the system; all we know is how many times the system has been used. If both types of information were available, it would be possible to examine patterns of individual usage. How many use the system? Of those, what proportion are casual users, and what proportion are serious or heavy users?

Figure 4

Use of IMPRESS Data Files

(Report Period: 6/28/70 - 4/19/71)

Data File			Date put on-line	Uses	%
PRES 68	--	1968 Election Survey - Michigan - SRC	09/17/69	3,216	36.1
DART 69	--	Dartmouth Coeducation Study - Winter '69	04/15/70	749	8.4
20 HS	--	High School Students - McDill, Meyers, Rigsby	12/30/70	684	7.7
DART 70	--	Attitudes to Coed Implementation - Dartmouth '70	05/09/70	642	7.2
X-POLITY	--	Banks & Textor's Cross-Polity Survey	06/24/70	503	5.7
YOUTH	--	Survey of Poor Youth in the United States	09/16/70	484	5.5
HOUSING	--	1970 Vermont Low-Income Housing Survey - Elites	10/27/70	418	4.7
ETHNO	--	Ethnographic Atlas - "Ethnography" version	03/15/70	294	3.3
NEW HAVEN	--	New Haven Local Political Study - Dahl	09/15/70	220	2.5
CLEVELAND	--	Cleveland Census Tracks - 1960 Census	10/16/70	185	2.1
U.K.	--	Almond & Verba 5 Nation Survey: United Kingdom	09/22/70	145	1.6
SOCPOL	--	World Handbook of Political & Social Indicators	09/14/70	135	1.5
PROF 70	--	Faculty Attitudes to Coeducation - Dartmouth '70	06/21/70	124	1.4
MOTHER	--	1968 Grafton County (N.H.) Maternal Care Study	09/14/70	116	1.3
PRES 48	--	1948 Election Survey - Michigan - SRC	07/06/70	100	1.1
MEXICO	--	Almond & Verba 5 Nation Survey: Mexico	09/25/70	99	1.1
DART 65	--	Dartmouth 1965 Panel Survey (3 waves)	09/23/69	98	1.1
GERMANY	--	Almond & Verba 5-Nation Survey: Germany	09/22/70	83	.9
ITALY	--	Almond & Verba 5 Nation Study: Italy	09/25/70	78	.9
U.S.	--	Almond & Verba 5-Nation Study: United States	09/21/70	75	.8
STATE 67	--	County-City Data Book - 1967 State Sector	09/15/70	67	.7
SERIES	--	Brookings Time-Series Econometric Model	06/21/70	65	.7
WORK	--	Organization of Work - Stanley Udy	02/15/70	59	.7
ABORTION	--	1969 Abortion Study - U. Minn., Duluth	02/10/71	48	.6
NATL SES	--	NORC Amalgam - SES & Vocabulary	11/09/70	43	.5
WINCHESTER	--	Winchester, Mass. - Stratification - Sokol	01/11/71	23	.3
CITY 67	--	County-City Data Book - 1967 City Sector	09/17/70	21	.2
SOKOL	--	Robert Sokol's Laboratory Manual	**	11	.1
PARISH	--	Baltimore Urban Parish Studies - Meyers	02/15/71	10	.1
CONVENT 68	--	1968 Pres. Nominating Convention - Scofield	03/10/71	8	*
LIFEWAYS	--	Data from "Lifeways" - Dartmouth College	**	7	*
ICPR PANL	--	1956-1958-1960 Elections (3 waves) - ICPR	02/10/71	3	*
CENSUS 60	--	1960 U.S Census - 1/1000 sample	04/19/71	0	
ENVIRON I	--	Environmental Consultants - N.Y. households	03/31/71	0	
ENVIRON II	--	Environmental Consultants - N.Y. individuals	04/07/71	0	
IBERIA-F	--	Iberian Studies (females) - linz - 1966	03/19/71	0	
Subtotal for IMPRESS*** & XIMPRESS***				8,813	100.0
Runs of EXPRESS*** exercises				121	
GRAND TOTAL				8,934	

---

* Less than 0.1%

Both SOKOL (put on-line 04/01/71) and LIFEWAYS (put on-line 07/29/70) have been removed from the array of active files. The data in the SOKOL file are such that they are better retained as EXPRESS* exercises; the LIFEWAYS data need extensive editing before they can be returned to the on-line group.

Figure 5

Twelve-Month Projected Cost of IMPRESS[a]

Nature of Cost	Inside	Outside	Commercial
3,022 hours of terminal time[b]	$ 9,066.30	$ 10,577.35	$ 21,154.70
190,305 C.P.U. seconds[b]	18,078.98	20,933.55	47,576.25
User scratch-file storage[c]	13,500.00	17,700.00	18,000.00
Storage of 35 data files[d]	39,602.97	51,923.89	52,803.96
Software storage[e]	3,065.98	4,019.84	4,087.97
Rent 10 Model 33 Teletypes[f]	10,200.00	10,200.00	10,200.00
12-month cost	$ 93,514.23	$ 115,354.63	$ 153,822.88

[a] The full year (6/28/70 - 6/26/71) is projected from actual data for the first thirty-nine weeks. Specifically, the third quarter was repeated as the projection of the fourth quarter.

[b] The rates for using the Dartmouth Time-Sharing System (as of January 23, 1971) are, as follows . . .

	Inside	Outside	Commercial
terminal/hour (33,35)	$ 3.00	$ 3.50	$ 7.00
terminal/hour (37)	3.75	4.35	8.75
C.P.U./second	.095	.11	.25
1K storage/month	2.25	2.95	3.00

[c] It is projected that, on the average, each of 250 utilizes 2K of disk storage (for scratch files) for twelve months.

[d] The disk storage requirements for the thirty-five data files on-line in April of 1971 were found to be 1,760,132 (36-bit) words. For these 35 data sets, the average sample size was 1,812 observations, and the average number of variables was 223.

[e] In object form, IMPRESS consists of 116,280 words in ten files; the costs reported cover this storage only. As a matter of information, IMPRESS in source form consists of 59,256 words in sixty-two files.

[f] In round numbers, the cost of renting a Model 33 Teletype is $85 per month. Thus, 10 teletypes rented for twelve months would cost $10,200.

Second, the fundamental logic of the software is such that numerous analyses be carried out within each execution of the system. Unhappily, it is not known how many analyses are actually done per run. If the typical user is including 5.2 variables per execution, he must be generating more output than a simple frequency tabulation or a single scatterplot. What is he doing?

Third, it would be most helpful to those of us who are responsible for both maintaining and improving the system to do some market research. Once he learns how to use the system, does the typical user grow weary of the interactive nature of IMPRESS? Does the student user need capabilities which we have not yet provided? Are his sessions at the terminal as intellectually stimulating as I have been suggesting in a number of papers, or does the use of IMPRESS become as dreary and as mechanical as the science laboratory?

As of this point in time, our attempts to learn something about the activities of our users have done little more than point out the things we do not know. I want to share these frustrations with you in the hope that 1) you will start to gather some information about your own activities, and 2) so that you can profit by some of our mistakes.

Summary

It is crystal clear to me that the title of this brief paper is appropriate: we don't know what we are doing. Furthermore, I find it very hard to believe that we will be able to obtain the support we need for our activities from our university administrations and from funding agencies until we obtain a better picture of how our software is being used and how expensive that use is. Equally important, we are not serving each other very well by the assumptions of elegance we perpetrate on each other with respect to our activities. Finally, in order to significantly improve our software and operating systems and modes of computation, it is mandatory that we gather some data and document our failures as well as our occasional successes.

Figure 6

Summary of Several Course Evaluations

Please grade each of the following aspects of the course, using the familiar scale:

	E	D	C-	C+	B	A	AVERAGE GRADE
	Unsatisfactory	Poor	Somewhat below average	Average or better, but not superior	Very good	Excellent	where E = 0 D = 1 C-= 2 C+= 3 B = 4 A = 5
Classes	0	1	5	25	33	13	3.675
Exercises	0	1	3	21	37	16	3.821
IMPRESS	1	1	6	7	33	20	3.941
Course	0	0	0	10	47	21	4.141

FOOTNOTES

* This paper was made possible by funds granted to Project IMPRESS at Dartmouth College by the Carnegie Corporation of New York and by the Alfred P. Sloan Foundation. The assistance of David A. Chapin and James A. Davis is gratefully recognized.

1. Wouldn't it be refreshing to hear a paper relating how the introduction of computer activity demolished what was once a successful academic course! Does anyone get the least bit nervous about the total absence of such papers?

2. Individuals with good intentions may attempt to honestly tabulate the costs of campus computer activities, but the results are shaded by university accounting systems (it is often difficult to distinguish real dollars from "funny money"), a diversity of monetary resources (grants, projects, part-time appointments, space in buildings), and a host of other factors.

3. The remaining question is whether the individual has the right to suppress data at the level of the group when he is a member of that group. The issue may be more clear if the question is raised in a different context. For example, does the undergraduate in a specific university have the right to suppress the fact that X% of that collectivity are manifest homosexuals? What about the comparison of W.A.S.P. and minority-group grade point averages? Returning to computer usage, the analysis of group-level data does not seem as sensitive or as threatening as some other examples.

4. Edmund D. Meyers, Jr., "IMPRESS and Undergraduate Education in the Social Sciences", Proceedings of a Conference on Computers in the Undergraduate Curricula (Iowa City: Center for Conferences and Institutes, September, 1970), pp.8.23-8.29.

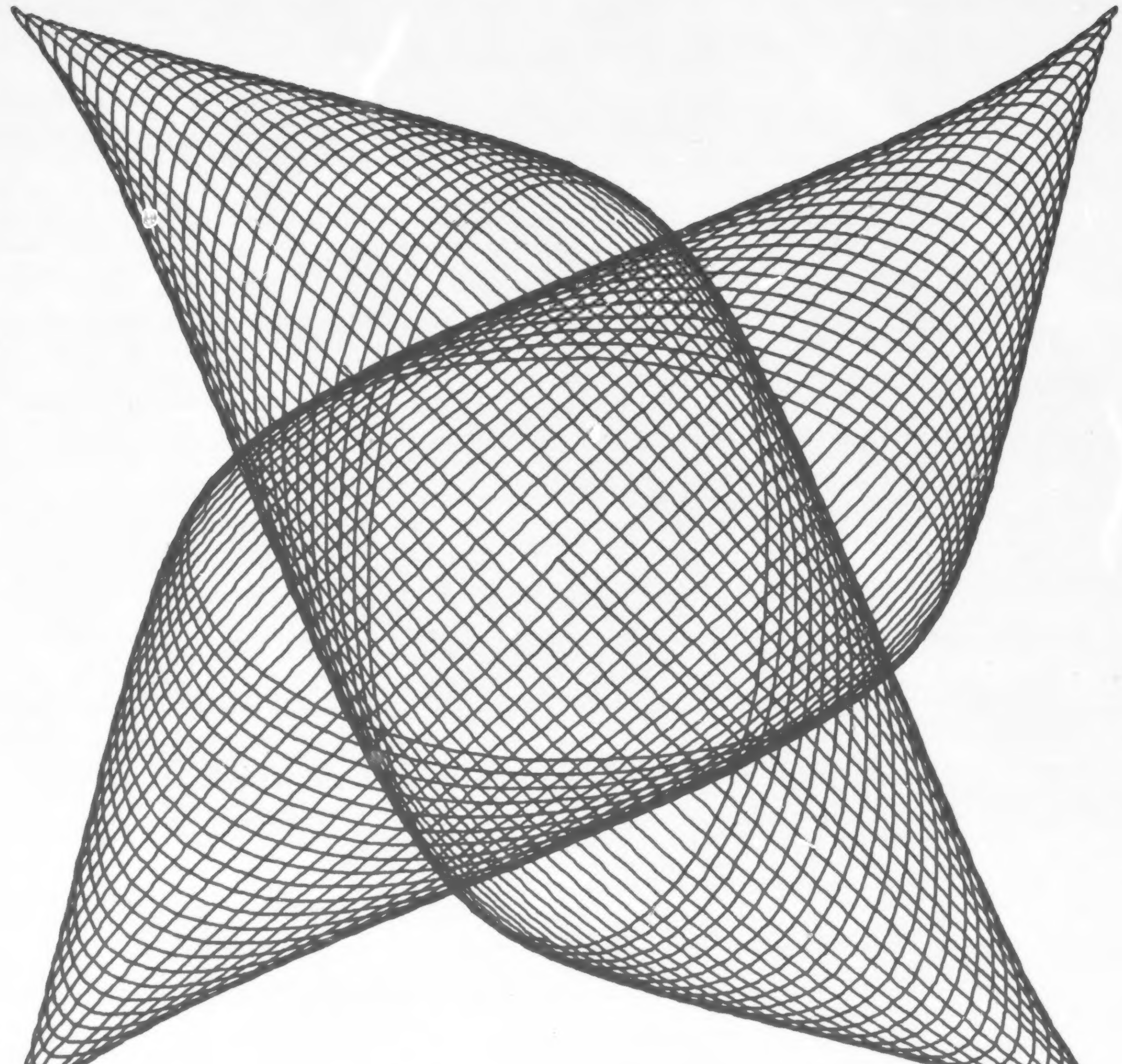
5. Early in the life of the project, software changes were introduced into the main system. It was quickly learned that undergraduates do not appreciate fluctuating capabilities; so XIMPRESS*** was created to provide a testing ground for new software which did not interfere with day-to-day usage.

6. Within DTSS, language BASIC has the following system-defined functions: DAT$ yielding mm/dd/yy, CLK$ giving hh:mm:ss, USR$ providing the user's account number, and TIM giving execution time in C.P.U. seconds. DTSS also provides the capability to write into a library file from any user number connected to the system.

7. Since the matching is done by a program, we not not look at the original data file. However, the security-conscious reader should be aware of the fact that it is possible for a member of the project staff to list and examine this file; as a function of the password protection scheme available in DTSS, the ordinary user *cannot* obtain access to this file.

8. Professor Campbell makes the same point with more fervor: "When the molecular biologist says that for every staff addition the university should expend $200,000 in laboratory space and equipment, and when benches in chemistry labs cost $1,200 per foot, should we sociologists thank God that we are not lesser men?" (Ernest Q. Campbell, "Recruiting Tomorrow's Sociologists", *The American Sociologist*, 5, February 1970, pp. 36-37).

9. Each student writes his name on the cover page, completes the course evaluation questionnaire, separates the cover page from the balance, and places each in a separate pile in the department office.



Stephen Toll

USES OF COMPUTERS IN THE TEACHING OF ECOLOGY

Robert W. Gill, Stephen H. Bryant, and Terry F. Mellon
University of California
Riverside, California

INTRODUCTION

Recent public interest in ecology has been reflected in increased undergraduate interest in the field. However, most of the theory and many of the applications of ecology require quantitative methods and models which pose difficulties for many beginning students. The purpose of this article is to show how these difficulties are being overcome through the use of computers in an ecology laboratory at the University of California, Riverside (UCR). The class, Introduction to Population and Community Ecology, is composed primarily of juniors and seniors, but many of the exercises could be modified for inclusion in the ecology portion of a beginning biology class for majors or for non-majors. Some of the exercises represent extensions of ideas presented by Justice (1967). The class uses IBM 2741 typewriter terminals linked to the IBM 360/50 computer on campus, and the programs are written in APL, an interactive, time-sharing language. This configuration was chosen for two reasons: 1) APL is fast, efficient, and easily learned, and 2) since the computer response is instantaneous, the student can be encouraged to try many variations on the model or method being used. However, campuses with less elegant facilities could lease, from one of the commercial suppliers, a terminal with a phone line linkup to a large computer. Alternatively, input for the programs could be obtained from the students during the prior laboratory period, providing time for routine batch processing and delivery of output by the nearest computer facility, prior to the start of the given laboratory exercise. Copies of the APL programs for the exercises discussed below are available from the authors if a 9-track, mini-reel tape is supplied with the request.

SPECIFIC USES OF THE COMPUTER

A) Demonstrating the Properties of Ecological Models.

Models of population growth, competition, and predation are difficult for beginning students to comprehend unless the models can be demonstrated in some way. Discussion of some of these models normally requires a knowledge of calculus and involves summations and matrix manipulations. Therefore, many students have difficulty following abstract discussions or numerical examples of the models. These problems can be partly overcome by use of a computer. If the general structure and properties of a given model are presented to the students in lectures or in their readings, the students can then be given representative data to use in a computer program that simulates the particular model. From the output the students are able to acquire a much better understanding of the strengths and limitations of the model and a better appreciation of the process or concept for which the model was constructed. If the students then supply their own input data and study the output, their understanding is further increased. Specific examples of models we have simulated or plan to simulate in beginning ecology classes at UCR are discussed below.

One simulation involves using the exponential growth model to predict future population sizes under particular schedules of survivorship and reproduction. With the computer program, successive sizes of the human population, for example, can be determined from the age-specific survivorship and reproduction data of the population and the initial age distribution and population size. Normally, the student is provided with raw survivorship and reproduction data for a particular population and an initial age distribution and population size. Real data for this exercise can be generated from the Demographic Yearbook, available from the Statistical Office, United Nations. The data are entered in the terminal and the computer calculates and prints out the age distribution and size of the population at successive intervals of time. The student is then asked to assume that certain control measures have been instituted in the population and is asked to adjust the survivorship and/or reproduction values accordingly. For the human population the assumption could be made that the pill was widely adopted and/or social attitudes toward family size changed, and reproduction in certain age groups was altered accordingly. The student then enters the new data and observes their effects on age structure and future population size.

The model could also be applied to other populations. Future population sizes of a particular pest species could be projected with and without the introduction of natural enemies or pesticides by assuming that such introduction would reduce the survivorship values in some particular way. Similarly, future population sizes of a threatened species of wildlife could be investigated before and after a program of habitat management and protection from deleterious effects of man, which would be expected to improve both survivorship and reproduction.

Through direct experience with computer simulations of exponential growth of populations the student acquires a more complete understanding of the process and a better appreciation for the magnitude of the changes needed in survivorship and reproduction to reverse population trends. Adequate discussion of the theory may be found in Connell et al (1970).

Another concept that is difficult for beginning students of ecology is the role of density in determining and limiting population size. Quantitative discussion of the topic normally involves little more than consideration of the very simple and unrealistic logistic model, in which the rate of growth of the population at any instant in time is determined by the initial rate of growth and the degree to which the population has achieved final population size (e.g.: Odum, 1959; Kormondy, 1969). The limitations of The logistic models have been adequately presented by Slobodkin (1961). We have modified the general method of Leslie (1945) for iterating successive population age structures and sizes by setting up survivorship and reproduction values that are functions of past population sizes. The data are summarized as a series of tables, the values of which are entered in the computer terminal along with an initial age structure and population size. The computer generates and prints out age distribxtions and population sizes for successive time intervals, using these data. The students are then asked to increase or decrease the intensity of the response to density by making appropriate changes in the tables relating survivorship and reproduction to previous population densities. The students may also vary initial age structure and population size. By observing the resulting pattern of age structure and population size, the students acquire a real appreciation for the possible roles of density in determining cycles and limits in population size.

Problems involving competition or predation can also be simulated for the student, providing him with a better understanding of these processes.

B) Eliminating the Tedium and Inaccuracies Associated with Performing Routine Calculations.

Many of the techniques and methods of ecology require numerous, repetitive calculations for their application to specific situations. Many students in ecology are dismayed by these time consuming calculations and lose track of the purpose of the exercise while performing the calculations. Furthermore, mathematical errors frequently occur, producing erroneous and misleading results. A properly written and tested computer program will eliminate these problems and greaty increase the value of the exercises for the students. An explanation of the methods and calculations involved in any particular technique can be incorporated in the program or presented by the instructor. The student then enters his data, receives the results of the analysis, and goes on to interpret them.

One such application of the computer involves determining the distributional pattern of populations of organisms in nature. Study of a population's abudance, behavior, and particular response to the environment normally requires knowledge of its spatial distribution. The distributional analyses used in ecology at UCR are the nearest neighbor method (Clark and Evans, 1954) and the I_δ method (Morisita, 1959). F. N. David's extensions of her diversity index (currently in manuscript form) also show great promise. These methods are applicable to the spatial distribution of terrestrial plants, attached aquatic organisms, animal nests or burrows, etc. The actual populations can be studied in the field, or photographs of them can be studied in the laboratory.

For the nearest neighbor method the distance from each organism (or its nest or burrow) to its nearest neighbor (or nearest nest or burrow) is determined for all organisms of the population in a particular area. The data are entered in a program that determines whether the distribution is regular, random, or clumped.

For the I_δ method a square portion of the field or photograph is divided into a series of small squares called quadrats, and the number of individuals (or nests or burrows) of the species in each quadrat is recorded. These data are entered into the terminal. The computer then doubles quadrat size by adding counts in pairs of adjacent quadrats. The procedure is repeated until the computer representation of the field consists of only two huge quadrats, each representing the sum of the counts of individuals in all small quadrats within it. The computer then calculates I_δ for each quadrat size according to the formula:

$$I_\delta = q\left(\frac{\sum_{i=1}^{q} (n_i)(n_{i-1})}{N(N-1)}\right)$$

where q is the number of quadrats, n_i is the number of individuals in the ith quadrat, and N is the total number of individuals in the whole set of quadrats. Finally, the computer plots the values of I_δ as a function of quadrat size. From this plot the students can determine whether the distribution is regular, random, or clumped. Furthermore, if the distribution is clumped, they can determine the average size of clumps, and whether or not the distribution within clumps is regular or random. The students then speculate as to the causes of the particular distributional patterns observed and design experiments to test their ideas. If time and equipment permit, they can conduct their additional observations and experiments.

The computer can similarly be applied in calculating statistics of population growth. A thorough presentation of the concepts of life tables, reproduction, and exponential growth of a population requires calculating net reproductive rates, intrinsic rates of natural increase, and mean generation times. Calculating these statistics involves taking sums of various cross-products, a time consuming procedure of uncertain accuracy when done by hand or on a calculator, but rapid and accurate when performed by the computer. Students can then compare the statistics for various real or hypothetical populations and relate the statistics to the corresponding changes in population age structure and size, determined under procedures outlined in (A) above.

A final example of this type of computer application is determining species diversity for a set of field samples. Most indices of species diversity are measures of the number of species present and the evenness of their distrubtion (Pielou, 1966). Species diversity of forest stands is required to evaluate their economic quality. Also, species diversity of many ecosystems changes drastically as a result of environmental changes, including increases or decreases in the amounts of many forms of pollution. The determination of indices of diversity from field samples is therefore of considerable importance to ecologists.

A commonly used index is the information statistic, H, which is calculated as:

$$H = \frac{1}{N}\ln\left(\frac{N!}{N_1!N_2!\cdot\cdot\cdot\cdot\cdot N_s!}\right)$$

where N is the total number of individuals in all species and N_1, N_2, etc. are the number of individuals in the first species, second species, etc. (Pielou, 1966). Students are able to enter into the terminal the raw data counts of numbers of individuals per species category in each sample, and a simple computer program calculates H according to the formula.

Similarly, the computer could be used to run statistical tests on the results of ecological experiments, to reduce various types of field data, or to fit theoretical distributions to observed distributions of organisms among sample units.

C) Introducing the Student to Systems Analysis.

This is the most exciting and powerful application of computers to teaching ecology and is still in its developmental stages at UCR. Through the use of computers the student is able to acquire a much better understanding of the application of systems analysis techniques to ecological problems, an approach that shows increasing promise (Watt, 1966, 1968). Furthermore, by designing and programming his own systems analysis models, the student is allowed to explore and extend his knowledge of ecology. Such application does, however, require that the student have effective knowledge of a particular programming language, as well as a good foundation in ecology.

The particular techniques and models of systems analysis to be presented depend on the interest and training of the students and faculty involved in the course. Very brief descriptions of three different applications of systems analysis techniques to ecological problems will be presented here.

A series of general models of ecosystems showing different degrees of complexity has been presented by Garfinkel (Garfinkel, 1967, Garfinkel, et al, 1964). The analyses reported by Garfinkel are incomplete, and the cases simulated are only a small number of those possible and desirable. A team of advanced undergraduates was assigned the task of extending Garfinkel's analyses, and the team achieved some progress.

A recent and relevant article by Harrison, et al (1970) presents the results of a systems analysis approach to discovering the distribution of DDT in ecosystems. Students interested in verifying the conclusions of this study could construct their own model ecosystem with the general properties of this model, enter possible values for the parameters of the components of the system, and simulate the system's subsequent behavior.

An older but somewhat similar model has been presented for the rates of decomposition of organic matter in natural forest ecosystems (Olson, 1963). Olson's model uses differential equations, which could be used directly, or which could be converted to normal difference equations and simulated.

CONCLUSIONS

As more models are developed for ecological problems and as more undergraduates and faculty receive effective training in computer programming and systems analysis, the potentials for use of computers in teaching ecology will increase greatly. The students produced using these types of approaches should be excellently prepared to meet the challenges of work with current ecological problems, either as technicians in research institutes or as graduate students in programs in ecology and environmental management in our various colleges and universities. In addition, other students who have been exposed to one or more of these uses of computers may be able to understand and appreciate some of the intricacy and complexity of ecological processes, be better prepared to evaluate the progress of ecology, and be critical of simplistic solutions to ecological problems.

REFERENCES

Clark, P.J. and F. C. Evans. 1954. Distance to nearest neighbor as a measure of spatial relationships in populations. Ecology 35: 445-453.

Connell, J. H., D. B. Mertz, and W. W. Murdoch. 1970. Readings in Ecology and Ecological Genetics. New York. Harper and Row. pp 4-17.

Garfinkel, D., R. H. MacArthur, and R. Sack. 1964. Computer simulation and analysis of simple ecological systems. Annals of the New York Academy of Science, 115: 943-951.

Harrison, H. L., et al. 1970. Systems studies of DDT transport. Science 170: 503-508.

Justice, K. E. 1967. The use of computer modeling in biology laboratory instruction. Experiment and Innovation: New Directions in Education at the University of California, 1(1):11-17

Kormondy, E. J. 1969. Concepts of Ecology. Prentice-Hall.

Leslie, P. H. 1945. On the use of matrices in certain population mathematics. Biometrika 33: 183-212.

Morisita, M. 1959. Measuring the dispersion of individuals and analysis of the distributional patterns. Memoirs of the Faculty of Science, Kyusha University Series E (Biology), 2(4):215-235

Odum, E.P., 1959 Fundamentals of Ecology. Philadelphia. W. B. Saunders.

Olson, J. S. 1963. Energy storage and the balance of producers and decomposers in ecological systems. Ecology 44: 322-331.

Pielou, E. C. 1966. Species-diversity and pattern-diversity in the study of ecological succession. Journal of Theoretical Biology 10: 370-380.

Slobodkin, L. B. 1961. Growth and Regulation of Animal Populations. New York. Holt, Rinehart and Winston.

Watt, K. E. F. 1956. Systems Analysis in Ecology. New York. Academic Press.

Watt, K. E. F. 1968. Ecology and Resource Management. A Quantitative Approach. New York. McGraw-Hill.

THE COMPUTER IN UNDERGRADUATE GENETICS

H. Malcolm Owen, Ph.D.
The University of the South
Sewanee, Tennessee 37375

The purpose of this paper is to show how the mini-computer, Nova, Data General Corporation, has become an integral, indispensable part of an undergraduate course in Genetics.

For many years, Biology 301, Genetics had been taught on purely traditional lines. Many texts have been used, primarily to vary the problems found at the end of chapters. Up until this year, approximately one half of the course consisted of classical Mendelian genetics, developmental genetics, DNA-RNA, etc. The last half of the course was devoted to population genetics, a subject of particular interest to the instructor, who is of the opinion that the majority of undergraduate textbooks are deficient in their coverage of this topic. Because of this deficiency, a disproportionate amount of time was consumed by both the instructor and his students on the sheer mechanics of deriving algebraic equations and problem solving by using the slide rule or desk calculator. The college received the computer in the summer of 1970, under a matching grant from the National Science Foundation. (1) Workshops were scheduled for interested faculty and students at the beginning of the fall semester. (2) Incidentally, this computer was restricted solely for educational purposes and is made available for students on a 24 hour basis. After instruction and study in the use of the Basic Language and with only several hours of practice, that section of the course, Genetics 301 dealing with populations was developed. That is, the study of the basic principles of Population Genetics was built around the use of the computer.

Two major objectives were accomplished as a result of bringing the computer into the course. First, the logic of deriving equations and the subsequent translation of these into "Basic", afforded an opportunity for understanding that had been missing in previous years. Many students in the past had simply satisfied requirements by memory, and ability to solve algebraic formulae. The computer programming proved to be a natural step to follow the derivation of an algebraic formula from a particular condition of, for example, the effect of partial selection in changing gene frequencies.

The following example, one of many, is cited to illustrate the manner in which the computer was used.

The instructor would furnish the following information on any given theoretical population:

	Genotypes			Total	Frequency of q
Initial Frequency	p^2,	$2pq$,	q^2	1	
Fitness	1 ,	1 ,	$(1 - s)$		
Frequency after Selection	p^2,	$2pq$,	$q^2(1 - s)$	$1 - sq^2$	$\frac{q(1 - sq)}{1 - sq^2}$
Relative Frequency	$\frac{p^2}{1 - sq^2}$,	$\frac{2pq}{1 - sq^2}$,	$\frac{q^2(1 - s)}{1 - sq^2}$		

The student was asked to: (1) derive an equation for Delta q when $0<S<1$; and (2) to submit a program in "Basic" that would enable him to calculate p and q at any given generation (t). Such questions as: At what value of q do you find the most rapid change? could be asked without demanding an unfair amount of the student's time.

Based on the relationship of $\Delta q = q_1 - q_0$ with any given selection coefficient (S) from 0 to 1, the following equation can be derived:

$$\Delta q = \frac{-sq^2(1-q)}{1-2sq^2}$$

This can be easily programmed as follows:

```
100 PRINT "COMPUTATION OF DELTA Q, RECESSIVE, 0 S 1"
110 PRINT
120 PRINT "VALUE OF S IS;"
130 INPUT S
140 PRINT
150 PRINT "VALUE OF Q IS;"
160 INPUT Q
170 PRINT
180 LET X = (-S*Q*Q)*(1-Q)/(1-(S*Q*Q))
190 PRINT "VALUE OF X IS;" X
195 REM "DELTA Q = X"
200 LET Q1 = Q + X
210 PRINT "VALUE OF Q1 IS;" Q1
220 REM "Q1 IS FREQUENCY OF Q IN NEXT GENERATION"
230 LET P = 1 - Q1
240 PRINT "VALUE OF P IS;" P
250 GO TO 120
260 END
```

In addition to the example given above, the following similar programs have been developed in Population Genetics (3):

1. Estimate of recessive frequency (q), variance (v), and standard deviation (σ), with and without Haldane's correction.

$$\hat{q} = \sqrt{R/N}\ , \quad Vq = \frac{1-q^2}{4N} \quad , \quad \sigma q = \sqrt{Vq} \quad , \quad \hat{q} = \sqrt{\frac{4R+1}{4N+1}}$$

2. Snyder's Test for Population Equilibrium.

$$S_1 = \frac{q}{1+q} \quad , \quad S_2 = \frac{q^2}{(1+q)^2}$$

3. Gametic Selection.

$$\Delta q = \frac{-sq(1-q)}{1-sq}$$

4. To calculate the number of generations required to reduce frequency of q to a given value if selection is slow.

$$sn = \log_e \frac{q_0(1-qs)}{q_n(1-q_0)}$$

5. Selection against diploid under the following conditions:

(a) If $s = 1$, $\Delta q = \frac{-q^2}{1+q}$

(b) Dominance, $\Delta p = \frac{-sp(1-p)^2}{1-sp(2-P)}$

(c) No dominance, $\Delta q = \frac{-sq(1-q)}{1-2sq}$

(d) Heterozygous advantage; Balanced Polymorphism.

$$\Delta q = \frac{Pq(ps-qt)}{1-p^2s-q^2t}$$

$$\hat{q} = \frac{s}{s+t} \quad , \quad \hat{p} = \frac{t}{s+t}$$

6. Migration.

$$(1-m)^n = \frac{q_n-Q}{q_0-Q}$$

7. Mutational Equilibrium.

$$(u+v)n = \log_e \left[\frac{(q_0-q)}{(q_n-q)} \right]$$

8. Calculation of Inbreeding Coefficient for any Generation.

$$F_n = \frac{1}{2N} + (1-\frac{1}{2N})\ F_{n-1}$$

9. Panmitic Index.

$$P_n = P_0(1-\frac{1}{2N})^n$$

10. Effective Population Size.

$$N_e = \frac{4N_f\ N_m}{N_f+N_m}$$

11. Rate of Growth of a Population.

$$N_t = N_0 e^{rt}$$

$$\log_e N_t = \log_e N_0 + rt$$

The computer, in addition to its contribution to understanding, secondly, has enabled the student to save a considerable amount of time that has been used for the preparation of de Finetti graphs and the solution of many more problems than ordinarily could be expected to be done.

The improvement of an undergraduate course in Genetics involving the use of the computer was not confined to the study of populations. The following programs were written for the students for the express purpose of saving time and increasing accuracy:

1. Calculation of Arithmetic and Geometric Mean, and Standard Deviation and Standard Error of Means and Difference in Means.

2. Chi-square, with and without Yates correction.

3. Chi-square Contingency Test.

4. Coefficient of Correlation and Regression Analysis.

These programs are extensively used by students, both in problem work and in the preparation of laboratory reports.

In summary, although the use of the computer is confined to the "Keyboard Mode" its positive value has been proved. It has enabled the instructor to accomplish an objective in approximately one-half the time with increased accuracy, interest and motivation.

REFERENCES

Li, C. C. 1955. Population Genetics. University of Chicago Press, Chicago, Ill.

Mettler, L. E. and T. G. Gregg. 1969. Population Genetics and Evolution. Prentice-Hall, Englewood Cliffs, N. J.

Strickberger, M. W. 1968. Genetics. The Macmillan Co., N. Y.

FOOTNOTES

1. NSF Grant GJ-1119 "Experiment in utilizing mini-computer facilities for instruction purposes"

2. Dr. Charles Peyser, Assistant Professor of Psychology and Director of the Computer Center has encouraged the development of projects of this type and is assisting both students and faculty in the use of the computer. His contribution is gratefully acknowledged.

3. A catalogue of programs is appended to this paper.

PROGRAM LIBRARY - BIOLOGY 301; GENETICS

Tape #	Manual #	Program
S01-200001	S00-200001	Chi Square
S01-200002	S00-200002	Standard Error of Sample Means
S01-200003	S00-200003	Chi Square with Yates Correction
S01-200004	S00-200004	Chi Square Contingencies Test
S01-200005	S00-200005	Standard deviation; Arithmetic Geometric Means
S01-200006	S00-200006	Standard Error of difference in means
S01-200007	S00-200007	Least Squares Regression
S01-200008	S00-200008	Estimate of q from R frequency
S01-200009	S00-200009	Snyder's test for equilibrium
S01-200010	S00-200010	Delta q, recessive $0<s<1$
S01-200011	S00-200011	Reduction of q in n generations
S01-200012	S00-200012	Delta q, recessive, $s = 1$
S01-200013	S00-200013	Delta p, dominance, $0<s<1$
S01-200014	S00-200014	Delta q, no dominance, $0<s<1$
S01-200015	S00-200015	Delta q when $W_2 = 1$
S01-200016	S00-200016	Replacement genes in migration force
S01-200017	S00-200017	Inbreeding coefficient, F
S01-200018	S00-200018	Panmitic index
S01-200019	S00-200019	Mutational equilibrium frequency rate
S01-200020	S00-200020	Effective population size
S01-200021	S00-200021	Delta q, haploid recessive, $0<s<1$

APPLICATIONS OF A WANG CALCULATING/COMPUTING SYSTEM IN THE UNDERGRADUATE GENETICS AND BIOMETRY LABORATORY INSTRUCTIONS

J. C. Tan
Valparaiso University
Valparaiso, Indiana

INTRODUCTION

In the last conference on computers in the undergraduate curricula held in Iowa City, several authors reported their experiences with small digital computers for undergraduate instruction in the physical and biological sciences (1,2,3). They have demonstrated that these small computers have a rather unique place in certain phases of undergraduate instruction because of (1) ease and speed of computer operation, (2) simplicity in programming, (3) low operational and purchasing costs, (4) greater maneuverability and space saving, and (5) greater student participation and interest.

This report describes the experiences with a WANG 360SE calculating/computing system with particular reference to the laboratory instructions in the introductory genetics and biometry courses, and the potential capabilities of such a system in the undergraduate biology curriculum.

EQUIPMENT AND INTERCONNECTING ARRANGEMENTS

The system installed in the Department of Biology at Valparaiso University is comprised of one 360SE simultaneous electronic package, four 360K keyboards, and a Model CP-1 card programmer. There are eight outlets installed in six different laboratories located on three floors, although a maximum of sixteen outlets is permitted. The keyboard and the card programmer can be plugged into any one of the eight locations; however, only four channels can be used simultaneously. The interconnecting arrangements thus provide greater flexibility and maximum usage of the system. The system cost approximately $7,000 at the time of purchase in September, 1969.

The Model 360K is one of the WANG 300 series solid-state electronic calculators. Its operational capabilities include addition, subtraction, multiplication, chain multiplications, divisions, reciprocals, $\sqrt{x}$, x^2, $\log_e x$, and e^x. Accumulation switches are provided for automatic accumulation of entries and/or multipliers. There are two adders which provide storage and recall capabilities. It has four storage registers whose capacity is 14-digit numbers and algebraic signs. The Model CP-1 card programmer is a compact, general-purpose, plug-in accessory which enables automatic calculator operations when used with any 300 series keyboard. It reads 80-step 2-digit octal program codes, corresponding to keyboard operations, from punched tab cards. Programs are prepared by tabulating the steps of a calculation, converting the key commands to key codes, and punching the prescored card (4). By using an IBM Port-A-Punch, for example, a 65-step program of 2 x 2 chi-square with correction for continuity requires less than three minutes of punching time. It is our experience that programming instructions and a short exercise of writing and executing the program can be completed within a 3-hour laboratory with an enrollment of thirty students.

The system requires minimal space. The electronic package (6 inches x 24 inches) can easily be placed on top of a laboratory cabinet. The keyboard (8 inches x 10 inches) and card programmer (4 inches x 10 inches) together require only 18 x 20 inches of table-top space.

The system is flexible enough to allow for future expansion and/or change of modules to meet the requirement of a specific application. For example, keyboards with trigonometric functions, the Model 362 keyboard with 26 storage registers, the CP-2 card programmer with ability to loop and branch, and other accessories can be added to replace an item in the existing system. The more advanced series is capable of such computer sophistications as looping, branching, executing subroutines, and manipulating arrays. It also has many plug-in accessories including powerful programming devices, extra data storage and input/output peripheral devices such as typewriter, teletypewriter, CRT display, on-line electronic input, etc.

TESTING OF GENETIC HYPOTHESES

Those who are teaching an introductory genetics course with laboratory will agree that most experiments cannot be completed without some form of data analysis. Students of genetics are constantly confronted with the testing of genetic hypotheses, from a simple 3:1 Mendelian ratio to a sophisticated quantitative genetics model. This means that in conducting an experiment, a student must do the following:

(1) formulate a genetic hypothesis,
(2) design an experiment,
(3) collect data,
(4) test the hypothesis, and
(5) draw conclusion of the mode of inheritance.

Two approaches are used in the testing of genetic hypotheses in our program. The first is a direct one: students are issued mutant stocks with known marker genes, for example, a white-eyed mutant and a wild-type stock of Drosophila melanogaster. They are told what to do in each step of the experiment and how to analyze data and draw conclusions. This approach is used to (1) acquaint the students with genetic techniques, such as culturing the fruit fly, and (2) to familiarize them with the procedures of collection and analysis of experimental data. As the students gain confidence and knowledge of genetic theory and techniques, an investigative approach is followed. Students are then issued coded stocks with marker genes unknown to them. The following represents a sample of topics for investigation:

(1) two X-linked genes
(2) two autosome-linked genes
(3) two balanced-lethal genes
(4) dihybrid with epistasis and linkage
(5) three X-linked genes
(6) one X-linked and one autosome-linked genes
(7) interaction between two genes

Students are furnished with information concerning what steps one must take to decode the unknown stocks, but they are not informed of what to do with their individual case. A decision is therefore involved in each step of the experiment. It is the purpose here to illustrate how a computing system like ours can be utilized to help students in performing computations and in testing genetic hypotheses of this nature.

Genetics is taught at the sophomore level for biology majors as a basic course in our core course program. It is offered also as an elective for other majors. Students at this level are characteristically not equipped with the background on statistical methods. Many of them even have poor attitudes toward and/or preparation for the application of mathematics in biology.

More often than not, they are so busy doing hand calculations that they lose sight of either the underlying genetic principles or the meaning of the statistical test itself. They spend too much time doing arithmetic and thus have very little time to think the problem through.

To remedy this situation, a series of programs has been developed with the following objectives: (1) to increase the scope of hypothesis testing, (2) to help the students to avoid repeated calculations, and (3) to aid students in making decisions in an experiment. Table 1 describes the title, nature, and computation time for the programs used. As indicated in Table 1, the time required for computing the various chi-square values for testing genetic hypotheses is drastically reduced. A student needs less than two minutes to finish computing the various chi-square values for programs 1-4.

There is another example of how this time-saving factor can benefit the class. Let us take the familiar case of genic interaction and segregation as shown in ears of corn. These ears represent a wide range of ratios and mode of inheritance. If the students are not told what is involved in the corn ears issued to them, they will have to investigate the particular case in a manner just like any geneticist will do to study the inheritance of a particular trait: They will first identify phenotypes, count the number of kernels in each, test the observed data against a genetic hypothesis, and finally draw conclusions on the mode of inheritance. Assuming only one keyboard and programmer is available and each student uses all four programs, a laboratory of thirty students needs about 45 minutes of computation time. This means that, in a 3-hour period, the instructor could have an hour to explain genetically what is involved in each type of corn, to pool individual data, and to point out the problems of sampling fluctuation and heterogeneity which might have led some students to reject a true hypothesis. The students will thus learn, on the spot, that hypotheses are never completely proven or disproven by a test of significance. The instructor may very well inject a thought-provoking question of what further genetic experiments should be carried out in order to prove that the hypothesis under test is genetically correct. Without the existing computing system, this type of laboratory work will probably take much longer time to finish, provided that the students have previous training in statistical methods and they are all very much quantitative-minded.

SIMULATION IN CLASSROOM

Many genetic phenomena cannot be illustrated through experimentation with living organisms in a 3-hour laboratory, or even in an entire semester; therefore, simulation comes into play. Many students, however, are not turned on by a simulated experiment. Aside from the fact that lively and colorful things are usually lacking in this type of experiment, the beginning students usually fail to appreciate the purpose and meaning of simulation and of mathematical modeling. The tedious calculations often stand as obstacles to the learning process. We have found that this situation can be somewhat improved by the use of the existing computing system reported here.

We have included in our course a simulated experiment on selection against recessive traits in a Hardy-Weinberg population. By the use of red and white beans, the experiment simulates a situation where two genes, A and a, have the same initial frequency in a population where random matings occur among AA and Aa genotypes, but none occur involving aa genotype. Students are asked to simulate this type of selection against the recessive trait with beans for twenty generations and record data. They are also asked to compare the observed with the expected genotypic frequencies after each selected generation, and the effectiveness of selection under different initial gene frequencies. Program 7, for example, is used for individual calculations, and graphs like Figure 1 can be made and discussed during the same period. Other mathematical modeling in biology, as well as population genetics, can also be done using the same procedure.

This type of simulation, though semi-automatic, achieves essentially the same end-effect with students, as it would in the case of teletype printout or television screen display. It may require a little longer time to finish a graph manually, but the graphic process itself may offer additional opportunity for comprehension and learning of the biological phenomenon.

AID IN BIOMETRICAL INSTRUCTION

Problems of teaching the statistical and mathematical applications in biology are essentially identical to those in a statistics course (7). Perhaps the situation is even worse in biology, because the undergraduate students generally lack such backgrounds, and the instructor has only limited time to present the topic. He is often compelled to teach, for example, the concepts of sample variance, standard deviation, standard error of the mean, and t-test within thirty minutes as the introduction to a laboratory during which this test will be used. The beginning students, on the other hand, often need a much longer time to comprehend, and to practice calculations, so that errors like a negative sum of squares from an empirical formula can be eliminated. However, we cannot afford overtreatment of these materials and reduction of time for illustration of the major biological concepts in a given laboratory.

Our experience indicates that the existing computing system can be used to help solve some of the problems involved in the teaching of biometry in order to achieve (1) reduction of calculation time, and (2) using it as a teaching tool. For example, using program 8, the $\bar{x}$, s^2, and s for a sample of 15 observations with 3-digit numbers can be calculated in one minute. The CP-1 card programmer has a STEP-AUTO switch located on the side panel. When the switch is on AUTO, the program steps are read automatically in sequence. When the switch is on STEP, only one step at a time is read (4). Although the STEP mode is originally designed for debugging a program, we have found that it can be used as a teaching tool, particularly when the program is written with this purpose in mind. Students can learn the step-wise calculations of a particular statistical or mathematical formula without having to do calculations by hand. They will also gain insights about programming this way. Many programs in statistics, mathematics, and other subjects for the 300 series calculators have been developed (8). These are satisfactorily adopted in our teaching programs.

SUMMARY AND CONCLUSIONS

1. The experiences gained with the use of the Wang 360SE system in undergraduate genetics and biometry laboratory instructions have been satisfactory.

2. The system's present and expandable capabilities, low cost, and simplicity in operation and programming are attractive features for undergraduate instructions especially when the big computers are not easily obtainable.

3. The potentials and advantages of small computing systems as instruction tools should be further explored.

4. The development of computer aided learning in genetics seems encouraging and the scope of such work should be broadened.

5. Computing techniques should be made a part of an investigative laboratory in biology.

LITERATURE CITED

1. Getzen, Forrest, W., *The Use of Small, Desk-Top Digital Computers for Undergraduate Physical Chemistry Laboratory Calculations*, pp 7.64-7.69, Proceedings of a Conference on Computers in the Undergraduate Curricula, The University of Iowa, Iowa City, 1970.

2. Child, W. C., Fimholt, J. E., and Ramette, R. W., *Experiences with the PDP-8 System in Chemistry Instruction*, pp 10.6-10.8, Proceedings of a Conference on Computers in the Undergraduate Curricula, The University of Iowa, Iowa City, 1970.

3. Spain, J. D., *The Use of the Olivetti-Underwood Programma 101 for Mathematical Modeling of Biological Processes*, pp 6.1-6.5, Proceedings of a Conference on Computers in the Undergraduate Curricula, The University of Iowa, Iowa City, 1970.

4. *Wang 300 Series Electronic Calculator Instruction Manual*, Vol. 1, Wang Laboratories, Inc., 1968.

5. Steel, R. G. D., *Elementary Mathematics for Statistical Analysis*, The Technical Press, Raleigh, 1962.

6. Kempthorne, O., *An Introduction to Genetic Statistics*, John Wiley and Sons, Inc., 1957.

7. Koh, Y. O., *The Computer As An Instructional Tool for the Statistics Courses*, pp 2.4-2.11, Proceedings of a Conference on Computers in the Undergraduate Curricula, The University of Iowa, Iowa City, 1970.

8. *300 Series Program Library Abstracts*, Wang Laboratories, Inc.

TABLE I

Title, Description, and Computing Time of Programs for Testing Genetic Hypotheses

Title of Programs[4]	Description	Computation Time[6]
1. Goodness of fit test	Chi-square test of the form paired Oi and Ei are entered and X^2 value displayed.	25 sec.[1]
2. $r_1:r_2$ ratio test	Chi-square test (5) for an expected ratio of $r_1:r_2$. Only the observed numbers and the ratio are entered.	15 sec.[2]
3. Test of independent assortment	Chi-square test (6) and only the four observed numbers are entered.	30 sec.[2]
4. Test of independence and linkage	Partitioning of the total X^2 into 3 component X^2-values (6). Only the four observed numbers are entered.	20 sec.[3]
5. Three-point test cross	The 8 observed numbers are entered and program computes for single cross-over percentages, observed and expected double cross-over percentages, map distances, and coincidence.	35 sec.[3]
6. Gene frequencies with co-dominance	Program computes gene frequencies and the expected genotypic frequencies under H-W equilibrium.	20 sec.[3]
7. Selection against recessives	Program computes gene and genotypic frequencies after n generations of selection.	15 sec.
8. Sample mean, variance, and standard deviation	Program records the number of entries, and calculates sample mean, variance, and standard deviation.	60 sec.[5]

[1] With 4 phenotypes, on the average.

[2] 2-digit numbers.

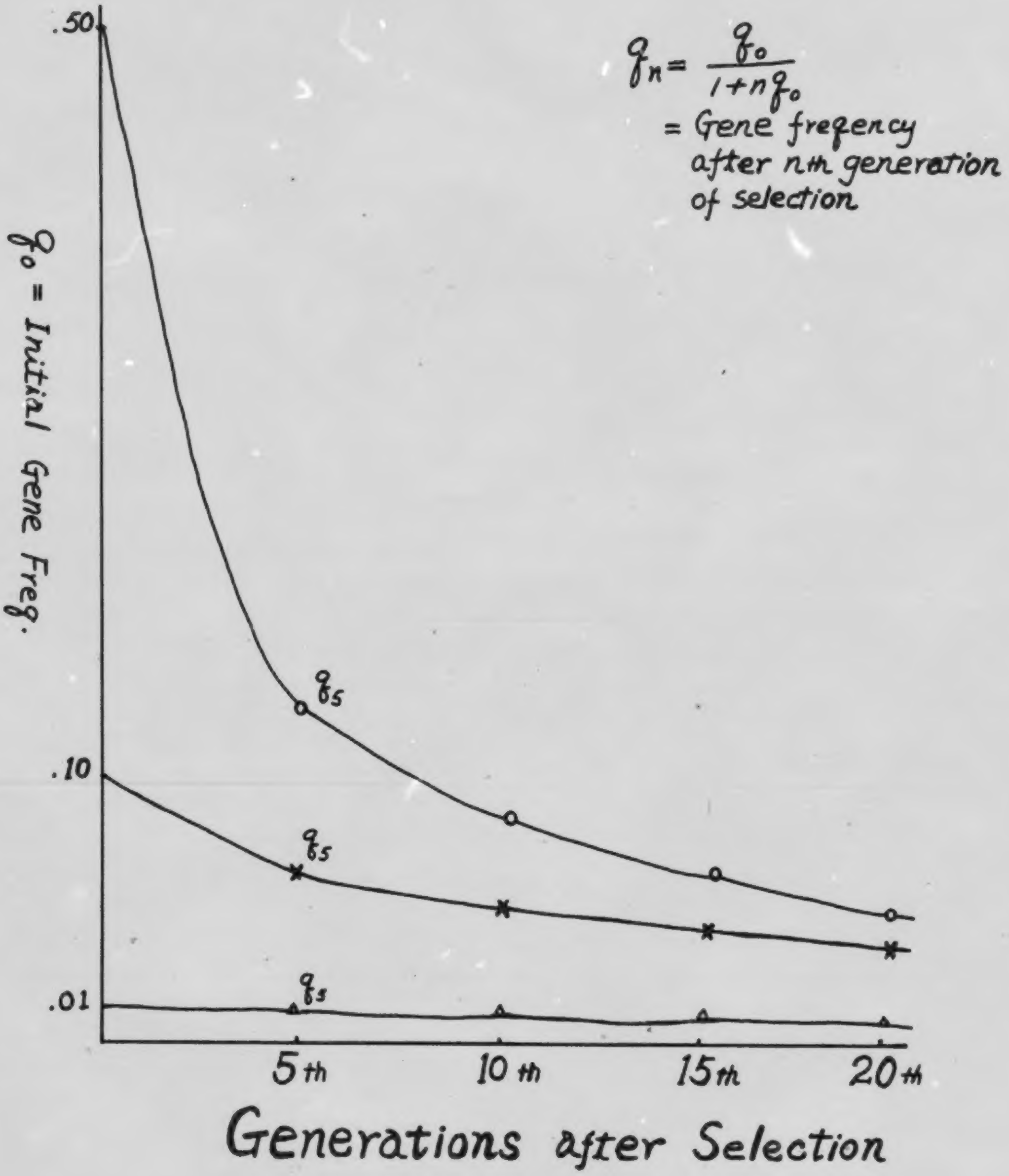
[3] 3-digit numbers.

[4] Tan, unpublished.

[5] 15 observations of 3-digit numbers.

[6] Time for indexing data and computing.

Figure 1 Selection Against Recessive



COMPUTERS IN THE UNDERGRADUATE CURRICULUM IN THE SCHOOL OF THE ALLIED HEALTH PROFESSIONS

Marion J. Ball
Temple University School of Medicine
Philadelphia, Pennsylvania

The recognition of the computer as a tool in the field of the Health Sciences has come to fruition. "The computer can act as a true extension of the mind if you worry more about getting jobs done and getting more people cared for than you do about who does it. There is no evidence that the stethoscope and typewriter kept us from being less true physicians than our fathers were, and we shouldn't assume that the new tools will make the next generation any less hukanitarian." (1)

> We need to think of ways in which a computing system not only provides a better approach to an old and recalcitrant instructional problem, but also of ways by which we can make more effective and efficient use of existing resources so that the total effect is to provide the student with more individualized instruction and the teacher with greater management power and scope. (2)

The accomplishment of this goal "...will require new interactions among medicine, the information sciences and the management sciences, and the development of new skills and attitudes on the part of policy-makers in the health-care system." (3)

Why computers as an aid to instruction?

> Their greatest asset is unquestionably the power to record, manipulate, remember, and respond to many persons individually and simultaneously.... are adaptable to an unlimited number of variations, and react only to the input of the student.
>
> Even at full capacity, it displays infinite patience, remains unemotional, and seldom requires rest. It can perform its role at any speed dictated...and can require specific answers or permit generalities ...and receive corrective and/or directive feedback without danger to the patient or the student ego.
>
> CAI requires less time to reach stated objectives than other methods. Moreover, CAI improves retention, helps identify individual difficulties, and allows students to progress at a self-determined rate. (4)

An example is the possible replacement of traditional multiple course examinations with "self examinations" whereby a student interacts with a computer at the time he feels he is sufficiently prepared. This method would, in fact, allocate "...the responsibility for academic achievement upon the student and permit course progression at individual learning rates." (5)

Although the advantages of computer assisted instruction far outweigh the disadvantages, there are several basic attitudes which are hindering the pace of computerization as an aid as well as a tool. In the case of the physicians, they have, as a group--

> traditionally cherished their ability to learn and retain large numbers of facts, to formulate a differential diagnosis and to carry on decision-making activities. Introduction of the computer into these processes could well be viewed by the doctor as devaluating his hard-won medical education and as undermining his intellectual contribution to medical care....Such a loss of status could have serious social, economic and political consequences for a profession that has historically enjoyed eminence in the public mind. (6)

Others feel that "...computer control of activities...promises to lessen further the need for highly specialized professional skills." (7)

Adversaries to computer assisted instruction (CAI) claim that "costs of both personnel and equipment are high and funds from both federal and local sources are being diverted to projects which offer more immediate results." (8) Time also proves to be costly as approximately 200 man-hours are required to produce one hour of computer aided instruction. Thus, many educators who are qualified to write CAI programs are relatively disinterested. (9)

"...Students...exposed to prototype computer-based consulting programs commonly express anxiety and displeasure..." (10) Perhaps the best way to overcome this attitude and therefore facilitate acceptance of computer assisted instruction by the student is to provide him with a course that eliminates the stigma of "computers" as an aid to education. Most dubious opinions are pre-judgments that are formed by hearsay as opposed to firsthand information. A course introducing computers and their capabilities and limitations has proven to be a valuable link toward widespread acceptance of CAI in all phases of undergraduate and graduate curricula. An example of such a course is explained below:

The course outlined in this presentation emphasizes several basic concepts which are carried through the entire program. Initially, our aim is to present the student with a broad conceptual feel for the possible uses of computers in the allied health professions. An example of where computer science has been applied in the health care areas is listed:

A. Anesthesiology
B. Bed Bookings
C. Billing Accounting
D. Blood Bank
E. Clinic Appointment Systems
F. Computer Simulation
G. Electrocardiogram (ECG) Analysis
H. Electroencephalographic (EEG) Systems
I. Food Service
J. Laboratory
K. Medical Library
L. Medical Records
M. Multiphasic Screening Systems
N. Patient History
O. Patient Monitoring
P. Physical Examinations
Q. Psychiatric Applications
R. Pulmonary Functions
S. Radiology

In addition, two comprehensive categories:

1. Total Hospital Information Systems
2. Regional Health Data Banks

A listing of where such systems may be found in the United States is located in Appendix #1.

Secondly, the principals of systems analysis and the actual field experience of defining an existing system from initiation of data flow to data transfer and final reporting procedures is being carried out. It is the aim of the course that these steps be taken in the area of the students' interest (or specialty). It is also anticipated that the work performed by the student will be advantageous to the field in which he is studying.

We try to emphasize the fact that the student may contribute to practical daily improvements as well as to theoretical utilization. The concepts of flow charting and decision tables are introduced to the student at this point as the basic tools for systems analysis. (See Figures #1 and #2). By applying the concepts of systems analysis in the health care areas, the undergraduate has the unique opportunity to work in the environment for which he is preparing himself.

When discussing these systems in the undergraduate curriculum it becomes necessary to acquaint the student with a new set of terms and concepts. After considerable experience, the following have become the basic information necessary to bridge the gap between the computer scientist and the health professional. (An outline of the most frequently used terms as they apply to the undergraduate as well as the graduate health professional appears below:)

I. Hardware

    A. Central Processing Unit
    B. Peripheral Equipment

        1. Unit Record Equipment
            a. Keypunch
            b. Portapunch
            c. Card Machines

        2. Input Devices
            a. Punched Card Readers, Magnetic Tape Readers, Mark-Sense Readers, or Paper Tape Readers
            b. Cathode Ray Tube
            c. Keypunch or Card Punch Device
            d. Teletype
            e. Typewriter

        3. Output Devices
            a. Cathode Ray Tube
            b. Keypunch
            c. Printers

        4. Input/Output Devices

        5. Conversion Equipment

        6. External (Secondary or Auxiliary) Storage

II. Software

    A. Computer Languages

        1. High Level Languages
            a. COBOL
            b. FORTRAN
            c. PL/1

        2. Low Level Languages
            a. Assembly
            b. Machine

    B. Using Computer Languages

        1. Programming
            a. Systems Analysis
            b. Flow Chart
            c. Coding

        2. Program
            a. Assembler
            b. Compiler

    C. Input

    D. Output

III. An Alphabetic List of Terms Related to the Operation of a Data Processing System

A. Access

  1. Immediate
  2. Random
  3. Serial

B. Access Time
C. Analog
D. Batch Processing
E. Bit
F. Byte
G. Closed Shop
H. Core (Magnetic)
I. Debug (or trouble-shoot)
J. Dedicated
K. Digital
L. Disc
M. Down Time
N. Generations

  1. First, Second, Third, Fourth

O. Installation
P. Location
Q. Real-Time
R. Time-Sharing
S. Turn-Around Time
T. Turn-Key
U. User

When firsthand experience is not possible, the course has been designed to bring selected visual aids, i.e. video tapes, films, etc., into the classroom. This segment is carried out with the expertise of the medical communications department, which directs these demonstrations. In addition, experts in various specialties are invited to speak. For example, one lecture may be conducted by a systems analyst and another, by a community health specialist. The following pages illustrate how the above ideas were applied in constructing an introductory course on computers in medicine. (See Appendix #2 for additional information.)

CURRICULUM PLAN

January 19th
1. Introduction to the course and an outline of what will be covered and what will be expected to be presented.
2. Read on Larry Weed's philosophy on the problem oriented medical record.

January 21st
1. Discussion and comparative systems description: Film on the problem oriented medical record.

January 26th
1. Introductory lecture on "What is systems analysis?" presented by Mr. Jan Elliott, Manager, Hospital Data Processing.

January 28th
1. Completion of Larry Weed's theory and slide presentation of Massachusetts General Hospital's program.
2. Read paper assigned in preparation for next lecture.

February 2nd

1. Required reading will be discussed.
2. Film presentation of the Homer Warner system in Salt Lake City will be shown.

February 4th

1. Lockheed slide presentation and philosophy will be discussed.

February 9th

1. Lecture and tour of how a systems analysis on Hospital Registration was carried out and how it is being implemented. (Lecturer - Mr. Douglas Ware).
2. Reading assignment.

February 11th

1. Continuation of February 9th lecture.

February 16th

1. Lecture review discussion on assigned reading.
2. Assign reports to students on current literature.
3. Film presentation - Patient History Taking by Computer -- a film by Digital Equipment Corporation.

February 18th

1. Reports will be presented on the assigned topics.
2. Film presentation - EKG Analysis by Computer - a film by Cesar Caceres.

February 23rd

1. Tour and lecture of data collection techniques and how a service bureau is used; presented by Mr. Ken Burton and staff.
2. Read assigned article for next lecture.

February 25th

1. Continuation of February 23rd tour and lecture.

March 2nd

1. Introduction to dedicated systems.
2. Pharmacy unit dose system slide presentation, and discussion.
3. Read assigned article.

March 4th

1. Slide presentation on multiphasic screening, and discussion.
2. Read assigned article.

March 9th

1. Clinical lab film, and discussion.
2. Read assigned article.

March 11th

1. ECG video tape, and discussion; project explanation
2. Read assigned paper.

March 16th

1. Field Project - perform a systems analysis on assigned topics.
    a. Define the problem and write a brief outline as to how you think the system can be improved.
    b. Gather examples, names and documentation. Note: To be conducted by Mr. Douglas Ware (see details in Appendix #2).

March 18th

1. Continuation of project for March 16th.

March 23rd

1. Film presentation - Total Hospital Information Systems.

March 25th

1. Lecture on comparative systems

March 30th

1. Lecture on choosing a computer system.

April 1st

1. Final examination.

A course similar in nature to the one explained on the preceding pages, or one with modifications to suit the instructor, gives the student an outlook conducive to receiving computer assisted instruction. This enables him to fully benefit from an educational tool that is limited only by the scope of human imagination.

"It is evident that a reorganization of the curriculum, far greater than in progress today, will be necessary to accommodate the diverse needs of the new student population....The power of the information sciences is such that it will, without doubt, alter the fact of medicine, and we can ill afford to ignore this impending reality." (11)

REFERENCES

1. Weed, L. L.: "Technology is a Link, not a Barrier, for Doctor and Patient", *Modern Hospital*, February, 1970, p. 80.

2. Stolurow, L. M., Peterson, T. I. and Cunningham, A. C. (Eds): *Computer Assisted Instruction in the Health Professions*, ENTELEK, Inc., (Newburyport, Massachusetts 1970), p. 7.

3. Schwartz, W. B.: "Medicine and the Computer", *New England Journal of Medicine*, Volume 283, December 1970, p. 1257.

4. Wismar, B. L. and Christopher, G. R.: "Computers and Medical Education", *The New Physician*, Volume 19, August 1970, p. 668.

5. *Ibid.*, p. 669.

6. Schwartz, W. B.: "Medicine and the Computer", *New England Journal of Medicine*, Volume 283, December 1970, p. 1258.

7. *Ibid.*, p. 1259.

8. Wismar, B. L. and Christopher, G. R.: "Computers and Medical Education", *The New Physician*, Volume 19, August 1970, p. 669.

9. Wismar, B. L. and Christopher, G. R.: "Computers and Medical Education", *The New Physician*, Volume 19, August 1970, p. 669.

10. Schwartz, W. B.: "Medicine and the Computer", *New England Journal of Medicine*, Volume 283, December 1970, pp 1259-1260.

11. Schwartz, W. B.: "Medicine and the Computer", *New England Journal of Medicine*, Volume 283, December 1970, p. 1262.

BIBLIOGRAPHY

1. Caceres, C. A. and Barnes, D. R.: "Training and Education for Acceptance of the Computer Sciences", *Annals of the New York Academy of Sciences*, *166*:1038-1044, (December 1969).

2. Davis, L. S., Collen, M. F., Rubin, L. and Van Brunt, E. E.: "Computer-Stored Medical Record", *Computers and Biomedical Research*, pp. 452-469, (1968).

3. Garfeld, S. R.: "The Delivery of Medical Care" *Scientific American*, *222*:15-19, (April, 1970).

4. Haessler, H. A.: "Recent Developments in Automating the Medical History", *Computers and Automation*, pp. 24-27, (June 1969).

5. Miller, J. G.: "Computer-Based and Computer-Planned Continuing Medical Education for the Future", *Journal of the American Medical Association*, *206*:621-624, (October 1968).

6. Overstone, J. A.: "Computer Assisted Instruction in Undergraduate and Post-Graduate Medicine", *The Medical Journal of Australia*, pp. 487-491, (September 10, 1966).

7. Payne, L. C.: "The Basic Principles and Brief History of Computer Technology", *British Medical Bulletin*, *24*:189-193, (March 1968).

8. Schwartz, W. B.: "Medicine and the Computer", *New England Journal of Medicine*, *283*:1257-1264, (December, 1970).

9. Stolurow, L. M., Peterson, T. I., and Cunningham, A. C. (Eds.): *Computer Assisted Instruction in the Health Professions*, ENTELEK, Inc. (Newburyport, Massachusetts, 1970).

10. Weed, L. L.: "Medical Records that Guide and Teach", *New England Journal of Medicine*, I. *278*:593-600, II. *278*:652-657, (March, 1968).

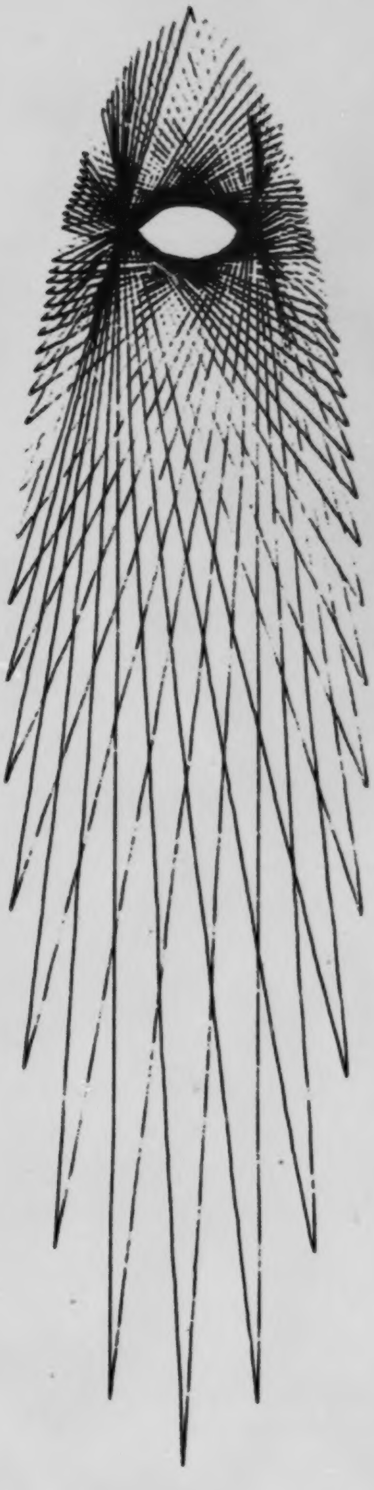
11. Weed, L. L.: "Technology is a Link, not a Barrier, for Doctor and Patient", *Modern Hospital*, pp. 80-83, (February 1970).

12. Wismar, B. L. and Christopher, G. R.: "Computers and Medical Education", *The New Physician*, *19*:666-669, (August 1970).

APPENDICES

Appendix 1. Operational Health Care Computer Systems (omitted -- contact author)

Appendix 2. Field Project - System Survey (omitted -- contact author)



William Carney

A GENERAL GRAPHICS SYSTEM
FOR COMPUTER GENERATION OF VISUAL AIDS*

Tim G. Kelley
Oregon State University
Corvallis, Oregon 97330

The instructional value of visual aids is commonly accepted among educators in all disciplines, as is evidenced by the wide variety of these materials that have become commercially available. Of the various visual aids media, it is probably safe to say that the movie is universally ranked most effective, both in conveying information and in holding the interest of its audience. This effectiveness stems directly from its unique capability for dynamically representing real processes. Yet even this medium suffers at times from its inflexibility and its inherent inability to function interactively. Essentially, the only option an instructor can exercise with regard to a movie is simply whether or not to use it, and once deciding to he has no further controls over the presentation, either as to content or sequential structure.

Not infrequently the inability to blend the film presentation in a more natural way with the rest of the lecture is sufficiently disadvantageous to seriously impair the value of an otherwise excellent film. There is a need in many instances for a compromise, for a visual aids medium capable of conveying information evolutionally while allowing the instructor to retain control over the specific content and timing of the presentation. It is the purpose of this paper to describe how the computer may be used to advantage to achieve such a compromise.

In our work we have been developing techniques for utilizing a time-shared computer facility in conjunction with a high speed, storage-scope computer graphics terminal for the generation of graphic visual aid materials. Our main objective has been to develop the capability of displaying computer generated graphics on-line, right in the classroom. This immediately suggests exciting possibilities for the infusion of an extemporaneous flavor into the demonstration; and given appropriately flexible supportive software this approach would surely optimize the interaction between the instructor, his demonstration, and the class. It has recently come to our attention, however, that there are other, surprisingly promising possibilities for exploiting computer graphics in the classroom besides on-line use and animated movies.

Specifically, we have found that the computer makes it possible and feasible to prepare sequences of 35 mm. slides which possess at least some of the dynamic qualities of movie presentations. Though not as dramatic as on-line computer displays, and certainly not as interactive, our experience indicates this approach to have remarkable potential for generating effective, individualized visual aid material. A discussion and illustration of this technique is to be the content of the present paper.

SLIDES?

The utility of lantern slides as an educational medium has been greatly enhanced by the advent of remotely controllable projectors, allowing the demonstrator to control the timing without the assistance of a projectionist. It is mechanically possible, at least, to display elaborate sequences of slides for which the continuity demands precise timing. The major difficulty with this concept is the excessive effort that has been required in the past to prepare such slides. An individual instructor must depend, to a large extent, upon existing slides prepared by someone else; but then he once again loses control over the content.

In those cases where the material to be presented is graphic in nature the preparation of slides falls into three phases: 1) design of content, 2) preparation of source material, and 3) photography. The several hours required at each stage plus inevitable delays of days or even weeks make this a very cumbersome process requiring a great deal of planning. In hindsight, the preparation of slide source material now seems an obviously fruitful application of computer graphic techniques. It turns out that the efforts required in phases 1 and 2 above are reduced orders of magnitude by the use of the computer, and even phase 3 is expedited considerably. It is quite possible to become "inspired" one morning, spend an hour or less at the terminal, and exhibit a sequence of 15 or 20 slides the next morning.

A DISPLAY LANGUAGE

It is one thing to say that a computer in conjunction with a graphic output device has the capability to draw pictures; it is quite another matter to claim that human time will actually be saved by having it do so. The programming required to generate computer graphics, though straightforward, is elaborately long and detailed. Quite obviously no instructor has the time to prepare such programs for each application. Furthermore, the utility of computer graphic techniques would be limited to just those very few teachers willing to learn to program reasonably proficiently, or to those who could hire a programmer. Most of us rely upon interested students for programming, and their skills have usually not yet reached the level necessary to do good graphics work.

In an effort to make computer graphics available to *any* instructor, regardless of his computer experience, we have designed an interactive computer graphics "language" or set of commands (which will be referred to as "tekcommands") to implement basic data manipulations and graphic operations automatically. With this command structure an instructor can often prepare his demostration without doing any programming whatever, and even at worst the programming required is the bare minimum necessary to generate the raw data (such as programming the solution to a set of differential equations, for example). Appendix A contains a list and brief description of the tekcommands currently implemented.(1)

The tekcommand structure is actually only part of an overall graphics software system capable of operating in three basic modes, which in turn correspond to three distinct levels of programming requirement. The most appealing of these, certainly, is one requiring no programming whatever on the part of the instructor, and we shall focus our attention primarily on this mode of operation.

Data generation when the system is operated in the "no-programming" mode is accomplished through an on-line formula translator, which we have dubbed "mini-tran". This translator allows mathematical functions to be defined on-line, expressed in nearly ordinary mathematical terminology. For example, an instructor at the graphics terminal might type

$$x(t) = x0 + v*t + A*\sin(w*t) \qquad \text{(Formula 1)}$$

defining a function named x to be a constant plus a linear term and an oscillatory term. The computer scans the expression for undefined parameters and then requests values for them, one at a time. In the above example the dialogue might proceed:

$x0 = \underline{1}$

$v = \underline{.75}$

$A = \underline{0}\ \underline{\text{by}}\ \underline{2}$

$w = \underline{2}$

where the instructor's responses have been underlined to distinguish them from the part written by the computer. As soon as the numerical input for one parameter is complete, the computer requests the next one by printing the name of another parameter (followed by "="). A free form of input is incorporated allowing these responses to be in integer, decimal fraction, or floating-point notation. The third parameter in the example, A, would be specially designated to be a varying parameter, meaning that under certain tekcommands its value will automatically be incremented, in this case by the value 2. Parameter variation allows the automatic generation of *families* of curves, rather than just a single curve. Families of curves are of particular value for instructional purposes because it is in visualizing the effects of varying a parameter that a student learns to appreciate that parameter's significance.

Mini-tran is, of course, just a special purpose compiler which generates code to evaluate the named function given the value of its argument. Designed specifically to be used in conjunction with plotting operations, it is nonetheless quite versatile within that framework. Quite a number of internally defined functions are available, such as the sine function referred to above. More than one function can be defined at a time, and, in fact, these can be pyramided, i.e., subsequent function definitions may refer to previously defined functions, their derivatives, or even their integrals. Mini-tran also has some branching capabilities allowing functions to be defined having different analytical form in different domains, or definitions to be made at special points. For example the definition

```
f(x) = N↑2 : sin(x) = 0 ; {sin(N*x)/sin(x)}↑2
```

defines f(x) to be $\sin^2(Nx)/\sin^2(x)$, but also takes care of the division by zero by assigning the value N^2 when sin(x) is zero. (For ":" read "if" and for ";" read "else.")

It will too frequently occur that mistakes will be made in typing formulas or responses. These are only an inconvenience if there is no audience, but in a classroom situation a most unseemly and unacceptable delay would often result. Error recovery can be facilitated by making use of various editing facilities, including special characters to cancel previous character(s), cancel previous line(s), restart the current sequence of inputs, return to a previous sequence, or carry out a character string "search and replacement" operation. The use of any of these error recovery facilities is only an optional convenience, however, and an instructor not wishing to bother to learn to use them can still get along quite nicely. This philosophy of "optional complexity" is present throughout the entire display system, and is very important because it means that productive results can be obtained with only a meager knowledge of the tekcommand language.

EXAMPLES

Numerous computer generated slide sequences have been prepared by several instructors utilizing the tekcommand language and formula translator. Excerpts from a few of these follow, with a brief description of the computer dialogue required to produce them.

The example formula presented earlier (Formula 1) was used to initiate a sequence for a very elementary physics course in an attempt to clarify the concept of derivative, or slope. After posing a half dozen conversational queries concerning plotting ranges, the computer displayed the full content of Figure 1. Placement of the curve label was specified by means of the "joystick," a graphic input device. The basic idea of the presentation was to build upon what was hopefully a common point of departure: the calculation of the slope of a straight line in terms of the coordinates of any two points on the line, as in Figure 2. The computer provided the coordinates of the points, again specified by graphic input, and then without special instruction evaluated the typed arithmetic expression for the slope. Several slides like Figure 2 were shown as a reminder of the arbitrariness of the choice of points, and to establish the limit of calculational accuracy expected, based on a slight fluctuation of the third significant figure.

The question under examination was how to operationally determine the slope of a non-linear curve, as in Figure 3, to the specified degree of accuracy. The obvious inadequacy of the previous method was illustrated by again choosing different pairs of points. Its success for a straight line, however, was supposed to suggest examination in a region so small that throughout it the curve would be straight, to within the specified accuracy, so that the choice of points would again not matter. To implement this procedure the tekcommand STRETCH was given to produce Figure 4, with the endpoints of the stretched region specified by means of the joystick. Two STRETCH commands later the curve appeared straight, as in Figure 5. But is it straight enough? Two more STRETCH commands established a value that further stretching would not influence, within the calculational accuracy, and then Figure 6 was displayed to sum up the presentation.

The point to be made here is not the merit of the pedagogy, but rather that the particular instructor involved could actually afford the time to produce this sequence himself, and could thus guarantee its effectiveness within his own pedagogical approach. This fairly elaborate sequence of some 16 slides was made at a total expenditure of about one hour of instructor time, which constitutes a tremendous saving over that required to prepare such a sequence by conventional methods.

The remaining examples are drawn from the physics area of wave motion and optics. Probably the single most difficult concept for the mathematically unsophisticated to grasp concerning these topics is the basic understanding of how a propagating wave is described mathematically. Figures 7 - 11 are part of two sequences designed to augment a discussion of this problem. The idea is to start by plotting a very simple function, x-vt, against x, with t serving as a varying parameter. This is simply a straight line of unit slope, independent of t, but its x-intercept is vt so that the propagation depicted in Figures 7-9 is clear. To generate the successive curves of a family, as shown here, the instructor must simply depress the carriage return key (CR). The almost obvious result obtained for x-vt is then relied upon to make reasonable the method for transforming an arbitrarily shaped function of x, such as a Gaussian, into a translating function of the same shape. We reason that a function depends only upon the value of its argument, hence replacing the static argument x by the translating argument x-vt must result in a new function which translates right along with its argument. Figures 10 and 11 are part of the sequence illustrating this.

These two figures also exhibit several features of the tekcommand language. A number of "effects" are controlled by the ON and OFF commands, such as the dashed lines and lack of tic mark labels, here. It is also seen that some curves of a family can be skipped over, which is accomplished by depressing the space bar a few times before triggering the next curve with CR. Finally, these figures entail the use of two sets of coordinate axis to compare different functions of the same variable. The tekcommand SPLIT is entered to divide the functions into the groups to be plotted on each axis set.

Figures 12-16 are excerpted from a sequence representing the reflection of a pulse on a string at a fixed boundary (vertical axis) and portray graphically the mathematical explanation of why the pulse is inverted upon reflection, in terms of the imagined superposition of two pulses so chosen as to insure satisfaction of the boundary condition while representing the incoming pulse in the physical region. The curves on this sequence of slides actually comprise a family parameterized by the time t but they can not have been triggered by CR because it would simply have added the next curve to those already present on the screen. The tekcommand CLEAR used here erased the screen and re-drew the axis before plotting the next curve. (The tekcommands all have names like STRETCH and CLEAR for mnemonic purposes, but for the economy of input demanded by classroom pressures they may be abbreviated by only their first letter.)

The superposition of two harmonics of the same frequency but different phase is another mathematically difficult concept at the elementary level. Here the trigonometric derivation is not so difficult but the physical meaning of the result is usually obscured by its complexity. Blackboard attempts to graphically clarify it invariably fail because of deficient artistry. Figures 17-19 are from a slide sequence which simulates the same blackboard demonstration, but with sufficient accuracy that the point is not lost. This type of sequence is particularly effective because it focuses the student's attention on just that part of the display he does not yet understand. (The full sequence continues for a complete cycle, of course.) The first step in generating such a sequence is to enter the LIMITS Tekcommand to establish axis limits which are different from the range of data requested. Successive segments of all curves (three, here) are then triggered by depressing the line feed key.

The superposition principle leads to the exciting phenomenon involving interference and the resulting optical intensity patterns, such as those produced by ruled gratings. To understand the physically realized multiple slit diffraction pattern requires combining the results for a grating idealized by infinitesimally thin apertures, like the four-slit pattern in Figure 20, with those for a single aperture of finite width A, as illustrated in Figures 21-23. The combined result is shown in Figure 14.

Figures 21-23 also exhibit two features of the tekcommand language. To sharply contrast the differences between a very wide and very thin single slit pattern, the slit width A, which had been assigned the increment 1, was changed from A=6 to A=20 for the third curve of Figure 22. Although this could have been accomplished by skipping 13 curves, or by changing the increment to 14 via the INCREMENT command, the most direct way was to simply assign the new value to A by typing A=20, as shown. The second feature exemplifies the versatility gained through the optional modifiers available with many tekcommands. Appending the integer modifier n to the CLEAR command, for example, causes N parameter incrementations to be made before plotting the first curve instead of the normal single incrementation. Thus the A=20 curve of Figure 22 was reproduced as the first curve of Figure 23 by typing C,0.

Figure 24, an example of a physically realizable four-slit intensity pattern, illustrates a special feature. The dashed "Fraunhofer envelope" had not been established as a function to be plotted, although it had been defined as a function. (When more than one function is defined the computer must be informed which ones are actually to be plotted, through the USE command.) To obtain an envelope on just one display, the command "Y=" was given which instructed the computer to "interpret what follows as a mathematical expression, compile code to evaluate it, generate and plot data, and then forget both the data and the code." The "Y=" command allows special "one-time-only" curves to be drawn, such as envelopes, limiting cases, comparison curves, etc.

Figures 25-28 demonstrate another technique for graphic representation, namely a density distribution of points. Because the appearance of density plots is quite like that of film exposed to light of the same intensity distribution, their use is especially appropriate here, providing a badly needed link between formal theory and physical reality. The "powder plot" mode of plotting, as we call it, is enabled by typing P (for POWDER) followed the first time used by the number of points to be generated. To revert to normal plotting, the command is simply repeated with no modifier. To superimpose a powder plot over an existing line plot, as in Figure 15, the JUMP command is used with the modifier 0. The JUMP command is similar to CR preceded by blanks, but with JUMP a non-positive number of curves can be "skipped".

Figure 16 contains the superposition of intensity curves for two wavelengths, demonstrating that in principle this method could be used physically to distinguish them, but Figure 27 brings home the difference between principle and practice that arises due to insufficient resolution. Assigning the value 20 to N, the number of slits, produced a drastically truncated intensity curve because the peak intensity goes as N^2 ; but renormalizing the data through the LIMITS command produced the nicely resolved doublet of Figure 28 (which in class was then followed by a powder plot representation).

SCOPE OF APPLICABILITY

As mentioned earlier, the source materials for these figures were produced via the formula translator, requiring no programming whatever on the part of the instructor. Unfortunately, the solutions to many problems of interest cannot be obtained in closed form. More complicated algorithms are often required which are beyond the scope of a simple expression evaluator. Yet the potential for providing a visual appreciation of the nature of solutions to more realistic models than can be solved analytically would appear to be one of the more powerful educational applications of computer graphics. This is especially true at upper division levels where study is often concentrated on techniques for obtaining approximate solutions to complicated problems.

To make the tekcommand stucture with accompanying data manipulations and graphic operations available to applications beyond the scope of formula translation, two other modes of operation are incorporated which permit interfacing with peripheral programs. The first of these conjoins mini-tran with externally defined subroutines which are provided by the instructor and which can therefore involve algorithms of arbitrary complexity. Issuing the XTERNAL tekcommand (X), followed by the function name, causes mini-tran to generate a transfer to these external subroutines rather than to the normally internally compiled code. The linkages are carried out in such a way that external subroutines need to be loaded only if the functions they evaluate are to be referenced. Other than a three line FORTRAN program, the instructor does not have to worry about these linkages. The external subroutine option is thus an extremely simple way to broaden the scope of the graphic system to any problem amenable to computer solution.

The second of the other modes replaces the formula translator altogether. The formula translator is cleanly interfaced with the Tekcommand-language implementation routines ("Executive" routines) which interpret commands, and perform the data manipulation and graphic operations. Whenever data is needed in any of the three modes, the executive routines simply make standard calls to an abstract "data generator". Replacing mini-tran as that generator therefore requires no modification whatever of the executive routines. The cost to the programmer in making such a replacement is that he must provide a somewhat more complicated interfacing than for the external subroutine method. What this buys, however, is the capability to develop entire instructional packages which can exhaust all of the facets and interesting aspects of a given topic.

An example will help clarify the idea. Such a package has been developed for the harmonically driven, damped harmonic oscillator. (2) In the interface was programmed a brief description of the problem, an array filled with the parameter names, and a call made to a standard routine in the tekcommand executive system to perform the input of parameter values, just as with mini-tran. The interfacing routine then permanently passes control along with arrays containing a description of the available options and corresponding specification codes to the tekcommand executive routines. These immediately display the options, which for this example were:

```
THE POSSIBLE TYPES OF RESULTS ARE:

    Displacement vs Time                   X VS T
    Velocity vs Time                       V VS T
    Both X and V vs T (one graph)          X,V VS T
    Both X and V vs T (separate)           X/V VS T

    Average Energy vs Driving Frequency    E VS W
    Phase Lag vs Driving Frequency         P VS W
    Both E and P vs W (one graph)          E,P VS W
    Both E and P vs W (separate)           E/P VS W

    Perpendicular Oscillators--non.int.    X,Y VS T

Enter your choice, symbolically:  X VS T
```

After selecting the type of result (underlined), the operation is identical to that for mini-tran, except it is often possible, as in this example, to provide a routine to compute appropriate limits for the dependent variable(s) from the parameter values, making it unnecessary to provide them during execution. It might be noted that the formats for plotted output are not really limited to those formally listed. The set of all mentioned dependent variables X,V,E,P and Y, here, correspond to the set of defined functions for mini-tran, so the same composition options obtain, through the USE and SPLIT commands.

A principal advantage of the third mode of operation is that, once developed, a package can contain sufficient flexibility that its results can be molded into a form to meet the individual needs of numerous *different* instructors, who may even be teaching at different levels.

The three levels of interfacing encompass a wide spectrum of options permitting the tekcommand language to be applied to obtain a graphic representation of the solution to any computer-solvable problem, regardless of complexity. Being provided with a flexible vehicle for transforming raw data into the appropriate graphic form, the instructor is relieved of much routine but time-consuming labor and can instead expend his efforts in deciding what the appropriate form is, in terms of effectiveness. This determination is itself greatly facilitated by the rapidity and ease with which different forms can be investigated. There appear to be virtually no restrictions as to level or discipline; any instructor presenting material which would be clarified by a graphic representation should be able to benefit either by interfacing his own problem solver to the tekcommand structure, or by using the formula translator directly. Students may also benefit in more direct ways. For example, many interesting problems require a graphic method of solution, but instructors tend not to assign such problems because of the inordinate amount of tedious labor involved. The ease with which complicated functions can be plotted using mini-tran completely removes this objection and makes it feasible, even, to obtain *families* of solutions to them. In conjunction with problem solving and other applications, a *very useful* feature is effected by the HARD COPY command, which causes a copy of all the graphic information currently residing on the terminal screen to be reproduced on paper.

It is too early to say whether the methods outlined in this paper will actually allow computer graphics to be utilized on-line as a classroom tool. We have found, however, that the computer makes it economically feasible to prepare very elaborate sequences of slides which do not just convey information, but which actually develop a theme or a process in at least a partially dynamic mode. New horizons for the instructional use of slide presentations appear when they are viewed from this new perspective. It may be worthwhile to enumerate the more important factors contributing to this enlarged view of their applicability. First, as mentioned, one may think in terms of developmental sequences rather than in terms of maximizing information content. Second, each instructor can feasibly generate his own individualized library of slide sequences which are then tailor-made to meet his own teaching requirements rather than someone else's. Third, new sequences do not have to be planned and preparation begun weeks prior to the date of intended use; the slide and lecture preparation can evolve simultaneously, guaranteeing a visual aid product which harmoniously supplements the pedagogical approach finally selected. (Unbelievable as it sounds, one does find himself actually preparing portions of his lecture while seated before the terminal!) All of these factors stem from two main advantages provided by the computer, when backed up by supportive graphics software: 1) The source material is many, many times quicker and much easier to design and generate than by conventional methods, and 2) it is much cheaper. (3)

ACKNOWLEDGEMENTS

The cooperation and assistance of the OSU Computer Center staff has been greatly appreciated, and particularly helpful were Mrs. J. Baughman and Mr. G. Rose of that staff. Numerous faculty members have contributed suggestions and criticisms which have been invaluable in shaping the design of the "language" to make it both useful and viable for them. Especially persistent in this regard have been Drs. C. Fairchild, D. Grillot, and N. Wise of Oregon State University, and G. Wolfe of Southern Oregon College. The active encouragement of Dr. J. J. Brady, Acting Chairman of the Department of Physics at OSU, and Dr. L. C. Hunter, Director of the OSU computer center, have strongly contributed to the progress of the work.

I am greatly indebted to Mr. M. Ebersole, presently a graduate student in computer science at Ohio State University. While an undergraduate at Oregon State University, Mr. Ebersole served on the programming staff at the Computer Center, but his contribxtions to the project far exceeded those of a programmer. The original compiler design for mini-tran and its interface with the tekcommand executive routines were developed independently by Mr. Ebersole, for example.

I especially thank Dr. N. Wise for reading and criticising the original manuscript, and Miss R. Williams for her patience and persistence in typing it.

I am also grateful to Tektronix, Inc. for the loan of a Graphic Computer Terminal (T-4002) and accompanying equipment, and to Mr. R. Hornicak of that company for his cooperative assistance.

REFERENCES AND FOOTNOTES

* This project was supported in part by NSF Grant GJ-51, NSF Aufendamp 370 Regional Center

1. An original version of the tekcommand language was discussed in "On-line Classroom Graphic Display," Kelley, T.G., to be published as part of the "Final Proceedings of the Conference on Computers in Undergraduate Science Education," edited by R. Blum.

2. This example is discussed more fully by Kelley, T.G., op. cit.

3. The 35 mm black and white negatives are mounted directly to produce black lines on a white background, so the cost of photography is between 5 and 10 cents per slide. Computer costs vary with complexity, of course, and with the extent of the effort devoted to the design of the sequence. Assuming only a reasonable percentage (~50%) of the time spent in "experimentation", typical computer costs per slide run between $.25 and $1.

APPENDIX: The Tekcommands

The modifiers shown on the following list of tekcommands have a standard meaning:

n	Increment the varying parameters n times (default = 1)
m	Suppress triggering requirement to plot m curves of family (default = 1).
i	Initial value of variable range (default: defined extreme).
f	Final value of variable range (default: defined extreme).
s	Step size of variable range (default: current value).
ℓ	Lower value of magnification range
u	Upper value of magnification range.
name	Either a function or file name.
ch	Name of a variable (default: independent variable).

Only the underlined modifiers are required (or the alternative equivalent graphic input).

Range Specification (ind. var.)

PLOT,n,m,i,f,s	Arbitrary relocation of independent variable range.
EXTEND,n,m	Erase screen, plot next segment.
(LF)m	No erase, plot next segment.
TOTAL,n,m,i,f	Combine segments (concatenation).
REAL TIME	Simultaneous multiple curve development.
SEGMENT = n.nn	Re-define segment size.

Family Determination

m(CR)	Trigger next m curves.*
JUMP,n,m	Jump to nth curve, trigger m curves.
CLEAR,n,m	Erase screen, trigger next curve(s).
INCREMENT = n.nn,...	Re-assign parameter incrementation values.
↑	Set parameter value, e.g., ↑P=5.0

* Prefacing the CR command with n blanks will cause n curves to be skipped. The CR and LF commands can also be combined. The general form is

(n blanks)(ℓ LF's)m

in any order, terminated by CR. The next n curves are skipped and ℓ segments added to the next m curves cyclically.

Magnification*

STRETCH,n,m,i,f	Magnification of independent variable (calc. data).
AMPLIFY,n,m,ℓ,u	Magnification of dependent variable (no data calc.).
ZOOM,n,m,i,f,ℓ,u	Magni. of both indep. and dep. variable (calc. data).
MAGNIFY,n,m	Magni. of both indep. and dep. variable (no calc. data).

* Graphic input can be used to specify ranges, eliminating the requirement for i, f, ℓ, u modifiers. MAGNIFY uses graphic input to specify center and input character for the magnification factor.

Function Definition

↑	Define function, e.g., ↑ Y(X)=expression.
USE,name	Establish function to be plotted. E.g., U,Y(X).
KILL,name	Clear named function definition.
XTERNAL,name	Establish external function definition, e.g., X,BESS

External Operations

FILE DATA,name	Save data on disk.
GET DATA,name	Retrieve data from disk.
HARD COPY	Copy displayed curves on paper.
QUIT	Terminate problem.

Utility

INFO	Review of Tekcommands.
DUMP	List parameter values.
VALUES	List pertinent problem values.
BAD PLOT	Erase and restore the screen contents.

Graph Detail

LIMITS,ch,i,f	Redefine graph limits for "ch"
OFF,effect	Suppress 'effects' as: AXIS, BARS, DASHES, GRIDS, LABELS, POLAR, MARKS.
ON,Effect	Enable 'Effect'.
SPLIT	Separates USE commands to divide functions into groups.

Computation and Labeling

(expression)=	Numerically evaluate expression.
Y=(expression)	Evaluate expression as fn. of ind. var. and plot.
Del Key	Initiate graphic input for labeling, vector drawing, coordinate specifications.

Figure 1

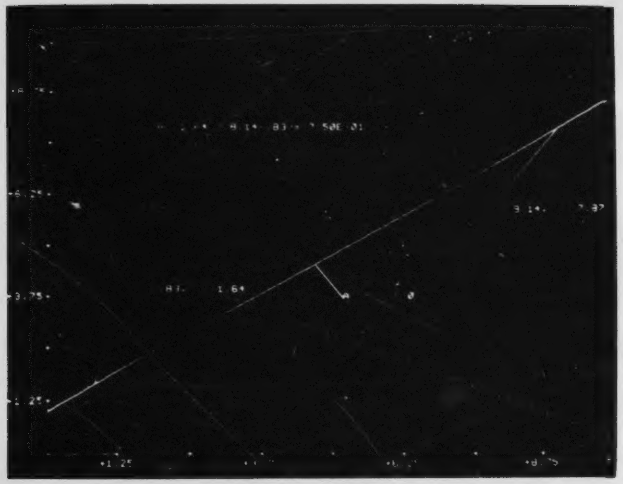


Figure 2

Figure 3

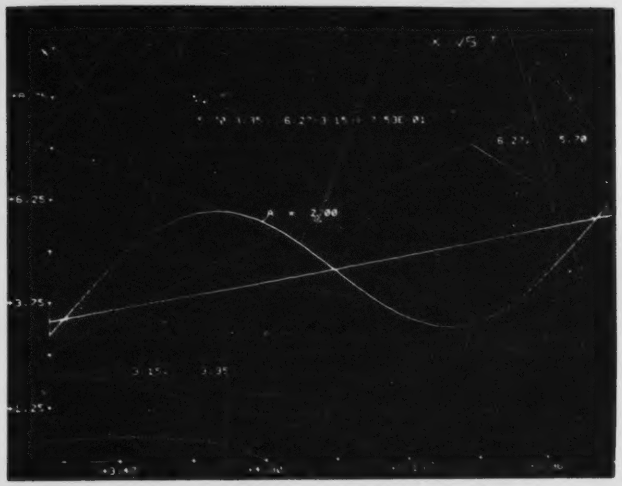


Figure 4

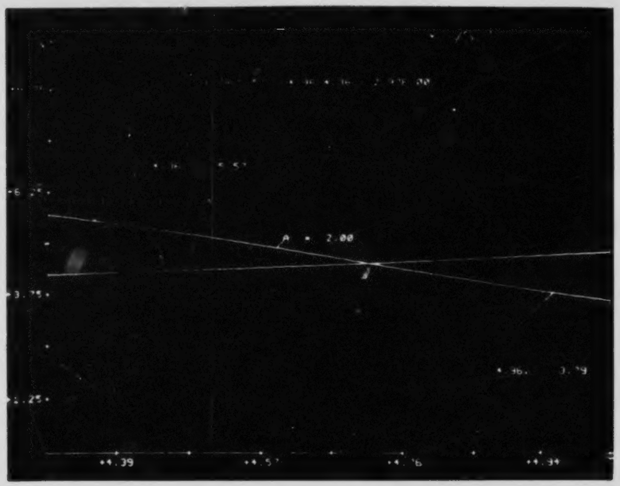

+6.25
+3.75
+1.25
A = 2.00
+4.39
+4.57
+4.76
+4.94

Figure 5

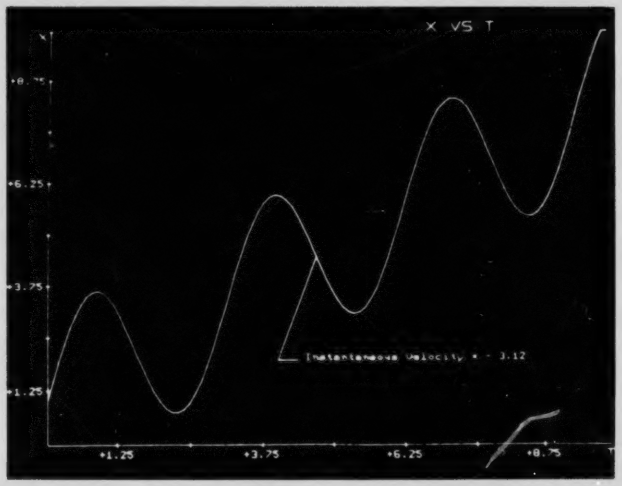


Figure 6

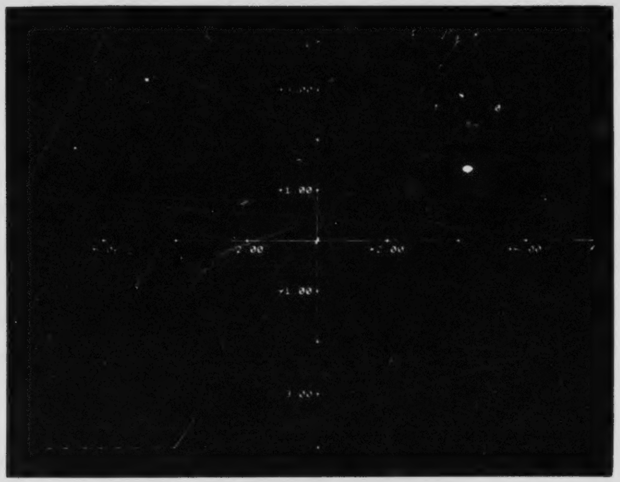

+1.00
-2.00
+2.00
-1.00

Figure 7

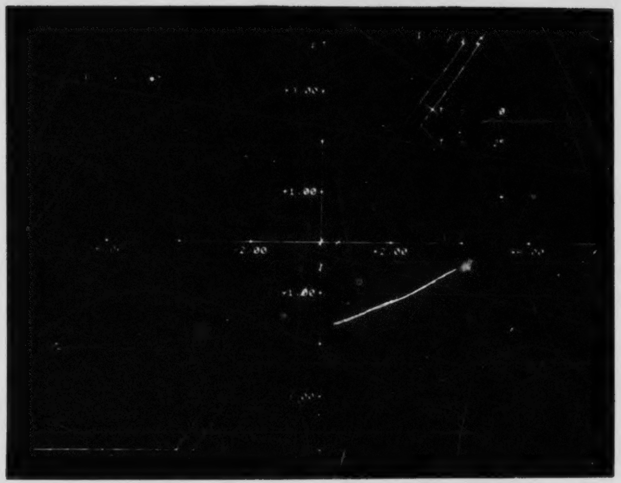

+1.00
-2.00
+2.00
-1.00

Figure 8

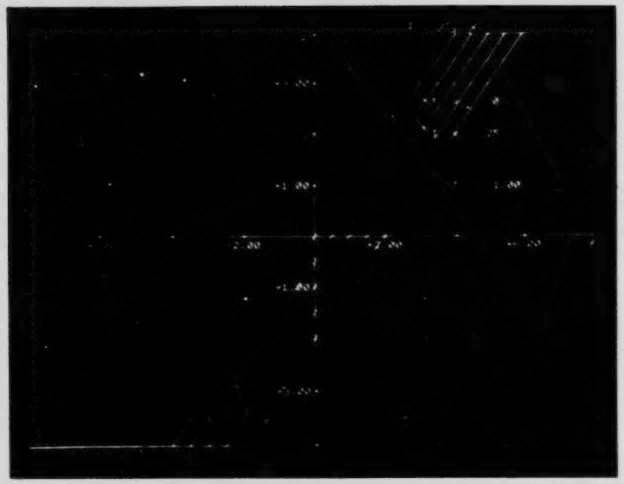


Figure 9

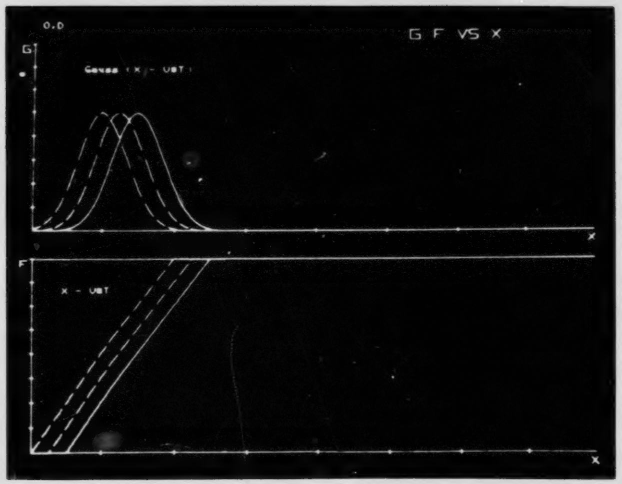


Figure 10

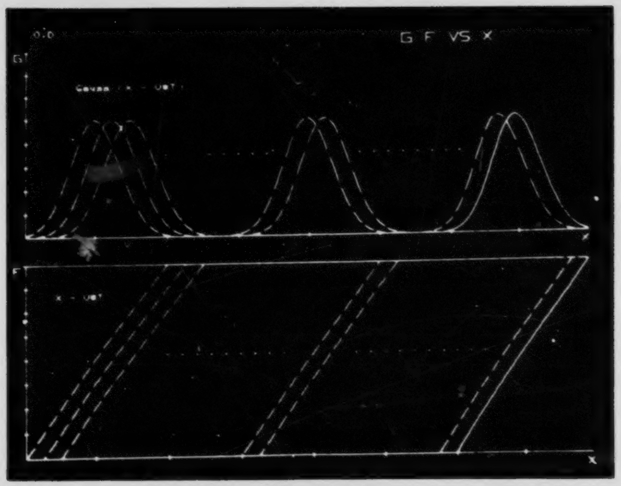


Figure 11

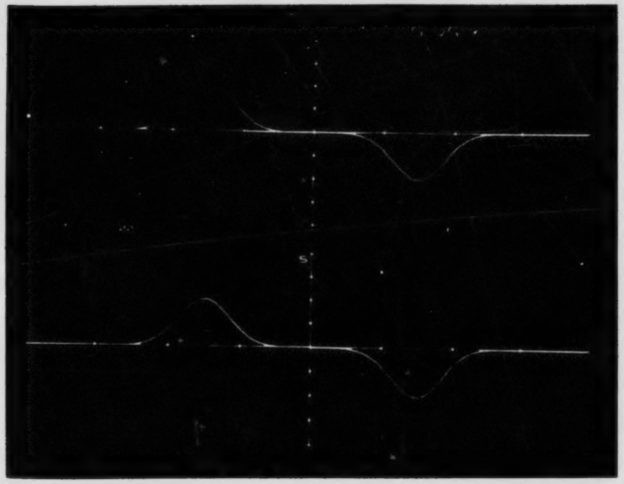


Figure 12

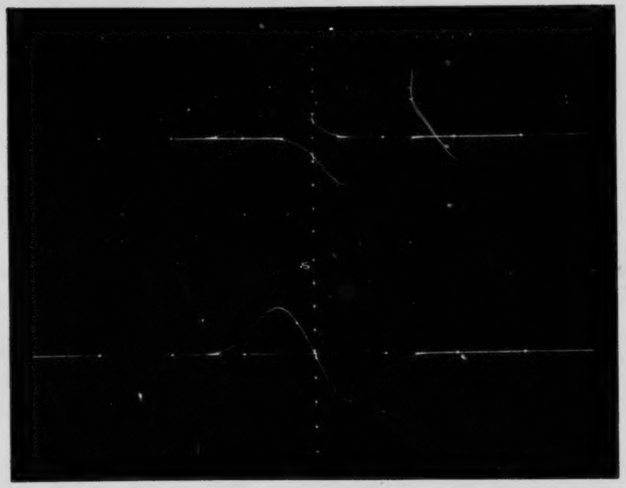


Figure 13

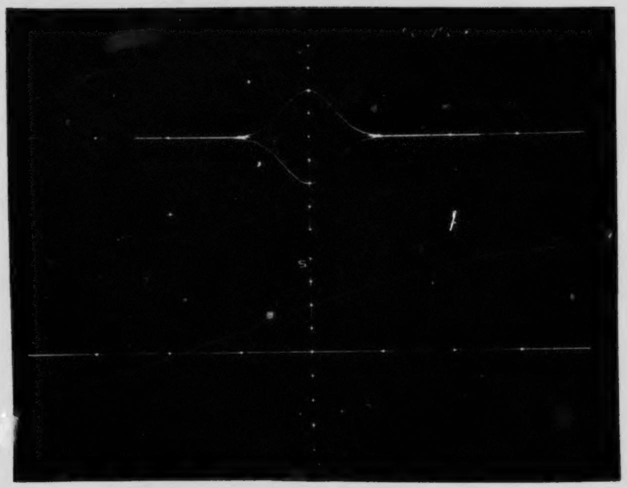


Figure 14

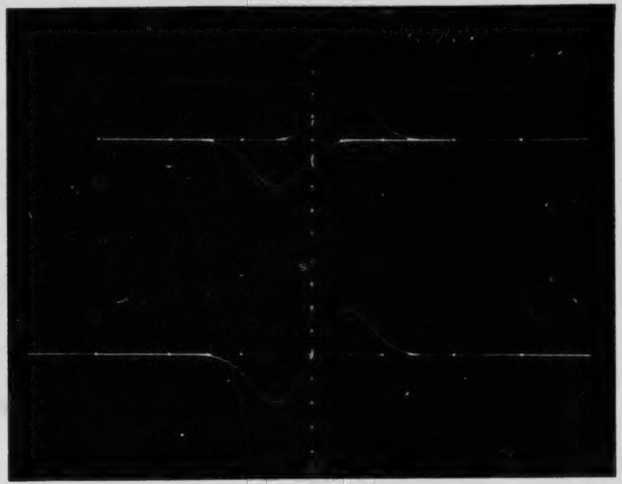


Figure 15

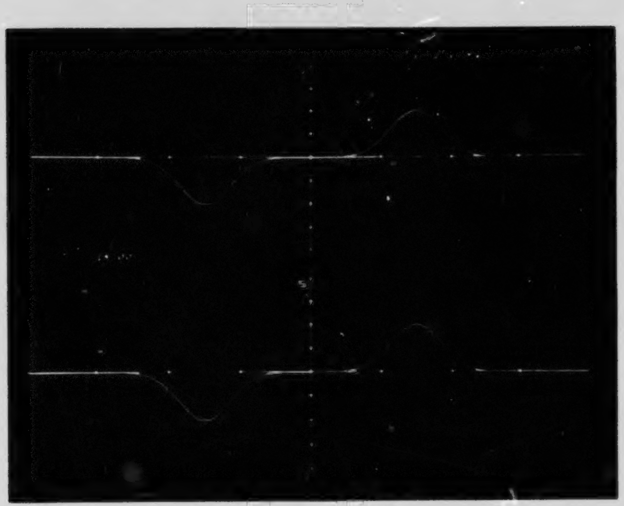


Figure 16

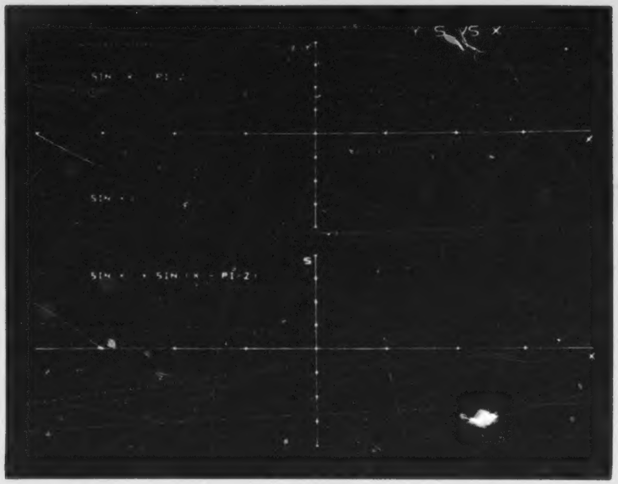


Figure 17

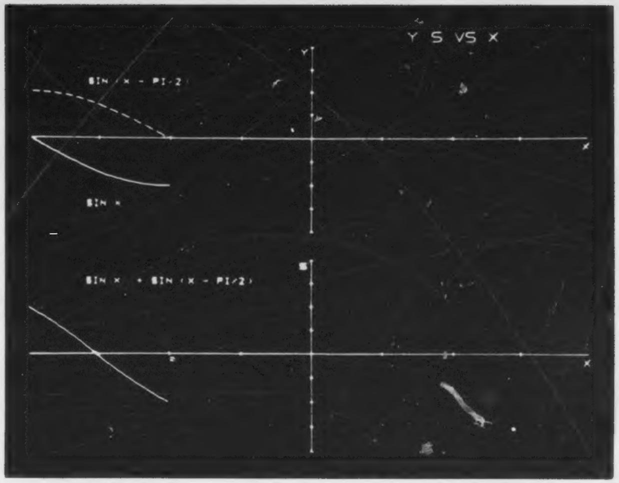


Figure 18

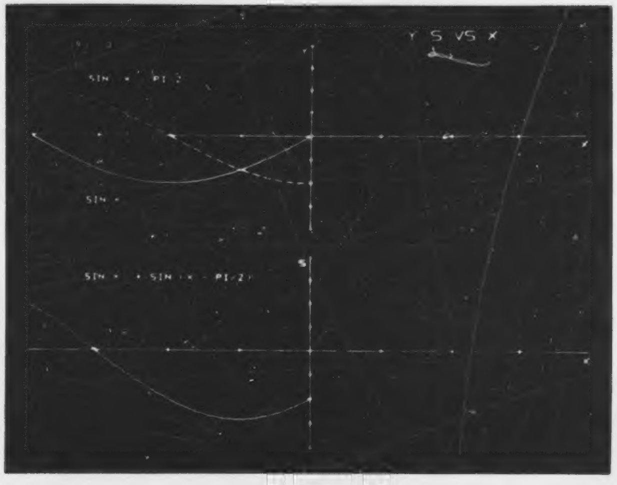


Figure 19

Figure 20

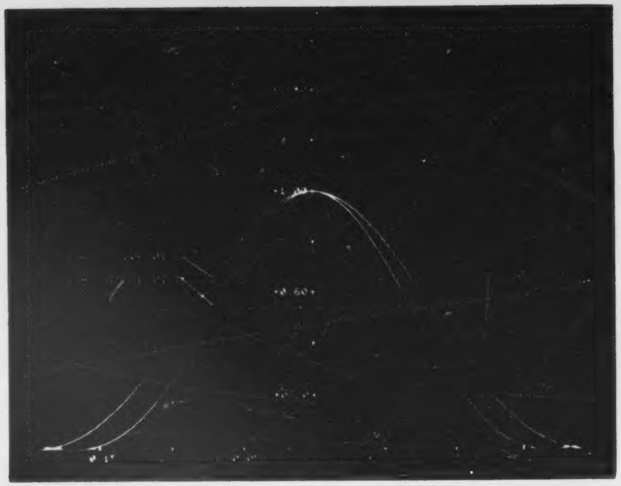


Figure 21

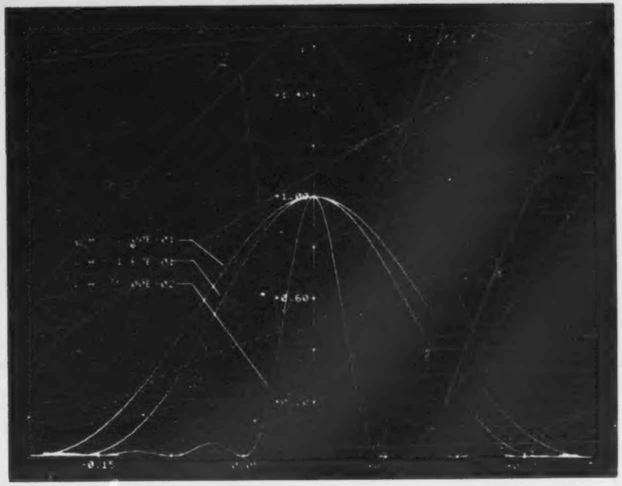


Figure 22

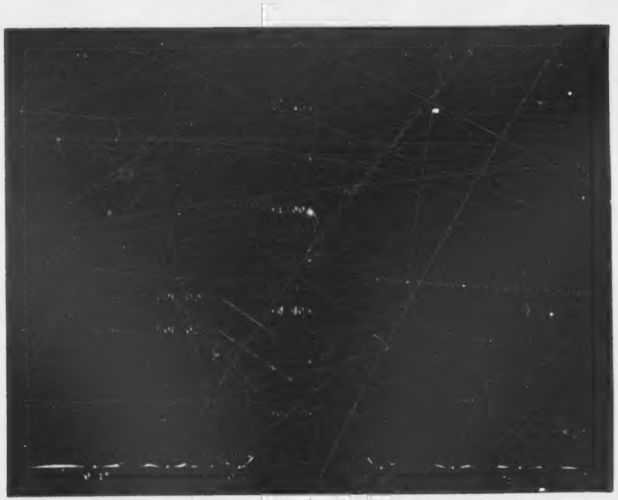


Figure 23

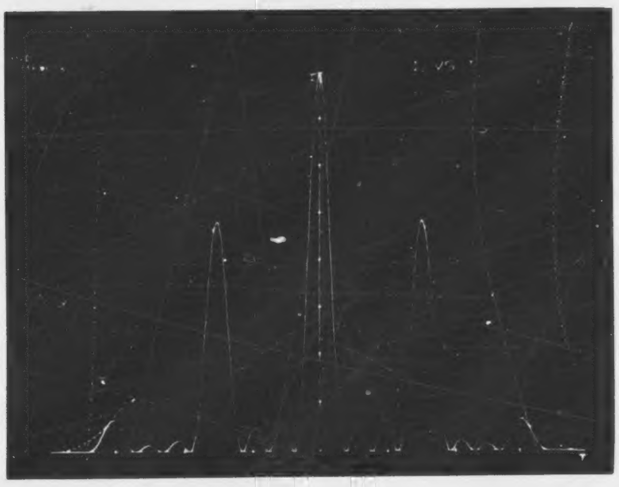


Figure 24

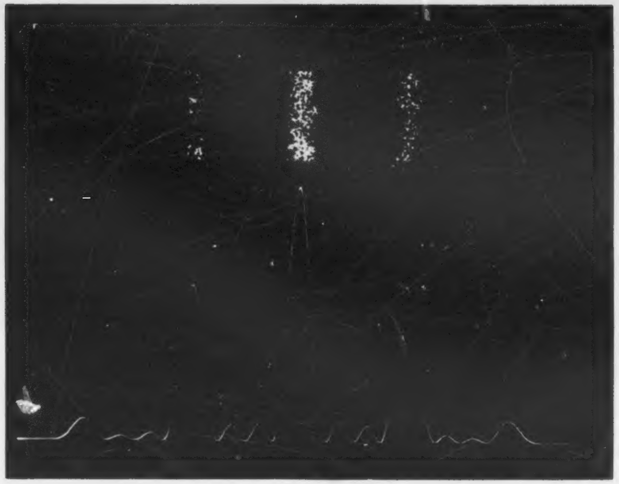


Figure 25

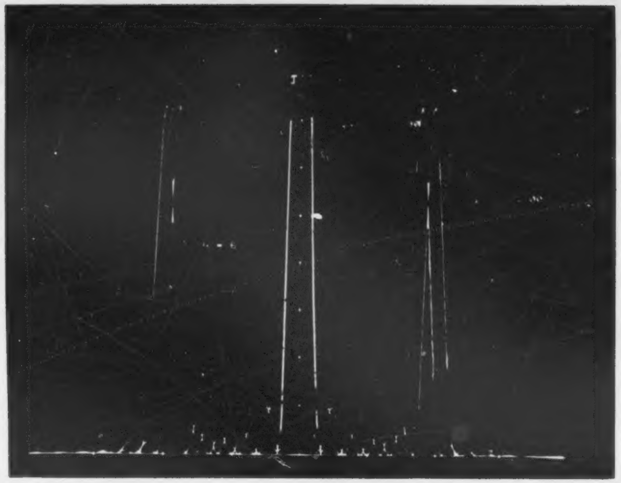


Figure 26

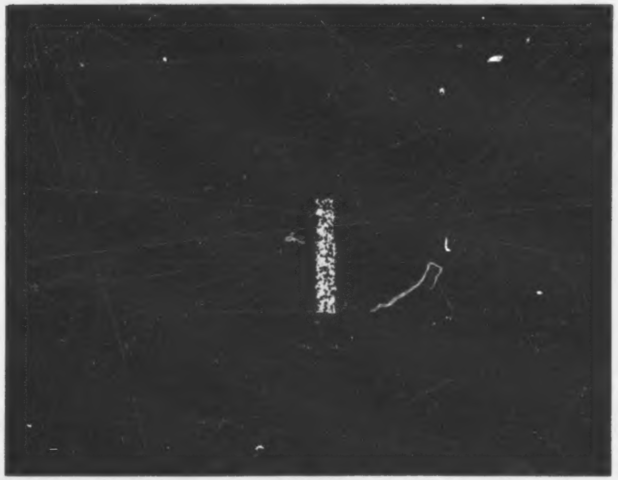


Figure 27

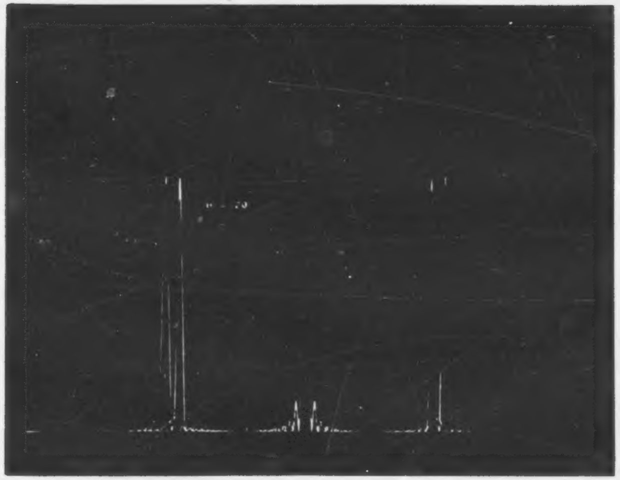


Figure 28

COMPUTER ANIMATED FILMS WITH A SMALL COMPUTER AND A SMALL BUDGET

Bruce R. Thomas
Carleton College
Northfield, Minnesota

Most computer animated films are produced using large computers, elaborate software systems, and expensive computer-to-film recording equipment. Last year Phillips and Geister* reported on the successful use of an output device with a bistable storage tube for the computer-film interface. However, they still retained the large computer and sophisticated software, aspects that make the process inaccessible to many potential users. This paper reports several further simplifications in the method that they detailed. These modifications should extend to a much greater audience the possibility of producing short lecture-expanding films.

We are producing computer-animated films using a small computer driving a storage tube device and recording the results directly with a standard Super-8 movie camera. As in the system described by Phillips and Geister, the storage tube plays a crucial role. It allows the small computer, working at a relatively slow speed, to generate arbitrarily complex frames, which are then recorded using single frame photography. This means that for any film that does not explicitly require a large memory (either a great quantity of text or a large amount of numerical data) the small computer can produce the same film that the more elaborate systems would, simply taking more time to do it.

Classroom Use

Two years ago we obtained a small computer with storage oscilloscope output capability. We envisioned using the facility in the classroom to provide graphical solutions of various problems in physics, and especially to illustrate the effect of changing the parameters of the problem. For instance, the time development of a wave could be depicted.

The intention was to program the computer to generate a display for a certain set of parameter values, then, either under automatic program control or under operator control, to change the parameters and generate a new picture. We soon discovered that although the equipment represented a tremendous improvement over traditional visual aids, it was not really fast enough to allow a very complete investigation of a problem in a reasonable amount of classroom time. The usual compromise was to show particular special cases of interest in class and to invite the students to pursue the problem outside of class using the program which had been developed.

We have now added, for a total expenditure of a little more than $100 plus about $5 per film, the possibility of producing film records of these programs showing a complete evolution of the parameters of interest. The use now takes the following form. We present the problem in class and look at a few specific cases with the computer as before so that the students understand the method of solution. Then the film is shown, presenting the solution with the parameters evolving continuously over the range of interest. This allows the full results to be presented in a few minutes in the classroom.

Finally, we still provide students access to the program so that they can investigate detailed quantitative aspects and values of parameters outside the range of the film. This is completely unfeasible with the more elaborate computer systems traditionally used for film production and, we feel, is an important pedagogical advantage of our simple system.

Hardware

This section contains a detailed description of our hardware, but we would like to emphasize that the storage oscilloscope is the essential feature. Any small computer with that output facility could be employed in a similar fashion.

The computer system consists of a PDP-8/L computer with 8K of memory, an ASR-33 teletype for input-output and a KV8/L graphic display system. One component of the KV8/L system is a Tektronix 611 Storage Oscilloscope. The KV8/L is driven directly by the computer and contains analog electronics capable of generating a vector between any two points of the oscilloscope in

4 milliseconds. In addition, it can produce shorter vectors in 250 microseconds, which is a very useful feature for rapid text-writing on the oscilloscope. All of the oscilloscope operations, including erase, are under computer control.

The photographic equipment consists of Vivitar Model 84 Super-8 camera with single frame capab'lity and an f/1.8 lens with 4-to-1 zoom. The zoom lens is not essential, but it does provide an extra degree of flexibility in the physical arrangement. The camera is simply clamped in front of the oscilloscope and the focus and zoom adjusted. We operate the unit in a darkened room rather than bothering to construct a light shield.

Interfacing between computer and camera consists of a very simple two-transistor monostable multivibrator with 0.5 second on time that actuates a solenoid. The solenoid literally squeezes the cable release which operates the single-frame mechanism of the camera. The circuit is driven by a pulse from an unused teletype output that happens to be available on our computer, but could easily be driven by tapping into the regular teletype line (the TTY can be turned off during filming) to extract a trigger signal.

Systems Software

We have available an interactive language with full graphic capabilities (a modification of the FOCAL language supplied with the computer). The only alteration that was necessary to do the filming was to write a new function subroutine to generate the filming signal. If the signal were taken from the regular teletype line, a simple output statement could be used.

Despite the somewhat slow execution speed of the interactive language, it is still especially convenient for two reasons. First, it allows much easier development of the program and adjustments of composition of the display. This aspect alone represents enough time saving to make up for the slower execution speed. Secondly, it leaves the program in a form that is readily accessible to students for further investigation. This includes interrupting the computer on a particular frame and interrogating it to find the current values of the variables, changing parameters and rerunning a portion, or actually altering a portion of the program, such as a numerical integration subroutine, to study the details of the method of solution. In cases where the speed of execution becomes a major factor, FORTRAN or various assembly languages are available. We have seldom found it necessary to use them.

Photographic Details

Fortunately, Kodak Tri-X Reversal film has recently become available in Super-8 format. This film has sufficient sensitivity to produce a satisfactory exposure of the oscilloscope screen using an ordinary camera at f/1.8 in the single frame mode.

We develop the film using standard negative processing, omitting the reversal process. This produces a very satisfactory black line on white background negative picture. Alternatively, it is possible to reverse the film to produce a white line pattern on dark background, but there seems to be no advantage in the extra effort and expense required. After editing, the films are loaded into Technicolor cartridges for convenient classroom use.

Applications

Our uses have included examples from classical and quantum mechanical wave motion, damped oscillating systems, antenna radiation patterns, and phonon dispersion curves. Generally, the method is appropriate for any display that can be presented with an interesting time dependent parameter.

The limitation that arises is the amount of computer time available for the filming. We do the filming in unattended overnight sessions when the computer would otherwise be idle. Therefore we have eight to ten hours of filming time. A 50' roll of Super-8 film contains about 3600 frames. This figure provides a very convenient scaling factor since each second spent in generating an individual frame results in an hour of filming time for the full roll. Thus we can afford to spend about 10 seconds generating each individual frame.

Given this 10 second restriction on drawing time, what is the limitation on how elaborate the picture can be? The figure cited earlier of 4 milliseconds per vector means that we could draw 2500 vectors in 10 seconds. This is obviously many more than would be required for a typical frame. Therefore the drawing time is not the limiting factor. Rather, the computational time that the computer requires to do the associated arithmetic is the essential consideration. This can be strongly affected by streamlining the program as well as by various efficiencies in numerical analysis. The net result is that the limit on elaborateness depends upon the cleverness of the programmer.

The film is viewed at 18 frames per second. If the 10 second frame generating time is taken to be representative of a typical frame, the filming process provides a time compression factor of 180. This allows a much more effective classroom presentation than was possible with the computer itself.

There are two obvious ways to relax the 10 second limitation and therefore to produce an even greater time compression. One is to use more than one overnight filming session to generate a film. The second is to plan a film that requires only a part of the full 50' roll. The full roll plays in about four minutes, but many interesting phenomena can be profitably displayed in as little as one minute or less. Ten hours of filming time devoted to a one minute segment represents a time compression factor of 600.

In summary, we have described a very simple, economical addition to our computer graphics system that allows us to produce computer-animated films. The method would be easy to implement on any computer that has a storage oscilloscope output. By using the films to supplement the direct use of the computer we have provided a significant improvement in classroom presentation.

NOTES

* Phillips, Richard L. and Donald E. Geister, "Production of Computer Animated Films from a Remote Storage Tube Terminal," Proceedings of a Conference on Computers in the Undergraduate Curricula, The University of Iowa, Iowa City, Iowa, June 1970.

COMPUTER ANIMATION FOR CHEMICAL EDUCATION

Fred E. Heidrich and Kent R. Wilson
University of California (San Diego)
La Jolla, California

Education is the communication of ideas. When one educates oneself through research, information is sought from others and from nature; when one is the educator, a means of passing information to another must be found. In either case, the effectiveness of the learning process is dependent on the competency of the communication link. Some ideas can be transferred through direct, real-time observation, others through mathematical formulas, words spoken or written, or pictures on a page. But many of the most fascinating areas of modern research simply cannot be adequately described through traditional means; they are too complex, or in mathematical form, too abstract. Thus new methods are needed to show students (and researchers) such hard-to-visualize concepts as the structure-function relationship in complex, biologically important macromolecules, the atomic motions associated with molecular collisions and chemical reactions, and the interactions of weather and traffic patterns which produce smog.

Since its invention the computer has helped the researcher clarify his own concepts of how nature behaves by translating abstract models and equations into tables of numbers. In order to educate others, the researcher then had to take the computer's output and put it in a form more generally understandable--usually printed words and graphs. While single computer-drawn graphs became common, hand-animated motion pictures were usually impractical because of expense and lack of facilities and training. Recent developments, however, in image-production hardwrre and software now make computer animation easily accessible for visual display of information.

The Senses Bureau of the University of California at San Diego--a group largely of undergraduates--was founded to employ computer animation both as an aesthetic and educational medium. Members of the Senses Bureau, in addition to the authors, are Watie Alberty, Noel Bartlett, Charles Morgan, and Stephen Pomeroy. Our goal is to produce films which combine information and beauty so as to appeal to a wide range of interests--to graphically display the beauty inherent in nature but often observed only by those who delve deeply. To us, education should be a dynamic, emotional involvement, both for pleasure and effectiveness.

To this end, we have so far produced two motion pictures. One, called "Patchwork 71", is designed to show the possibilities of computer animation, as well as to provide an interesting tour of many facets of chemistry. "Patchwork" is an anthology of highlights from the work of twenty computer animators, which we have edited, with the help of some magic from the optical printer, changing black and white into color and altering sequence and timing. The visual rhythms are matched by musical images from many composers on an analog computer of sorts--Moog Synthesizer. The chemical subjects included range from molecular quantum mechanics through the hydrodynamics of water released from a sluice gate. The image production devices employed range from very sophisticated (and expensive) oscillographic processors to a beautifully effective (and not so expensive) use of a felt pen and incremental plotter. We hope "Patchwork" will excite viewers both about computer animation and chemistry.

Our other film is one of a planned series of films called "A Protein Primer" for undergraduate education in chemistry and bio-chemistry. Proteins are a class of very large molecules which play a role in almost all bodily processes from metabolism to genetic information transfer. An amazing fact about proteins is that all of them are composed of only about 20 different basic building blocks called amino acids. How the hundreds of amino acids are arranged in a given protein determines the function of that protein. The subject of our first film is myoglobin, a rather small protein (composed of 153 amino acids) which acts as a storehouse of oxygen in muscles. We show the structure of amino acids, how they are linked together to form a chain, and how the three-dimensional winding of the chain specific for myoglobin is related to its oxygen storing ability.

The images which compose an individual frame are produced using a program called ORTEP from Oak Ridge National Laboratories. ORTEP is commonly used by X-ray crystallographers to produce ball-and-stick models of molecules. The input information consists of the co-ordinates of each atom as well as a description of the bonding between atoms. By varying the viewing position as well as the number of atoms shown from frame to frame animation of the molecules is achieved. On a CDC 3600 computer, a separate magnetic tape containing the image information is produced for each color appearing in the final version. The information on the tapes is then transferred into black-and-white film (Kodak Recordak Dacomatic)--a separate strip for each color-yusing a 35mm pin-registered camera on a 4060 graphical computer generously made available to us by DatagraphiX of San Diego. The black and white film strips are then overlaid through appropriate color filters onto a single 16mm color master. The soundtrack, composed by Gino Piserchio on the Moog Synthesizer, is then added optically.

Our animation process is only one of several methods using computers to produce films. The technique an individual employs depends, of course, on the equipment available to him as well as the subject he wishes to display. "Patchwork" and "A Protein Primer" demonstrate that computers are now fully capable of producing aesthetic educational films.

Note: While this abstract is in written form, we intend to present our actual contribution as computer animated films in color and sound.

Stephen C. Harvey

THE USES OF COMPUTING IN A MODERNIZED PROBABILITY AND STATISTICS COURSE

Edward L. Spitznagel, Jr.
Washington University
St. Louis, Missouri 63130

This paper describes what I believe to be a rather thorough integration of computing into a one-year probability and statistics sequence at Washington University. The students in the course are typically juniors and seniors with a mathematics background of at least two semesters calculus and little to no computing experience. The organization of the course had been discrete, combinatorial probability in the first semester and rather weak normal statistics in the second semester. This year it was decided to teach a course with rather different emphasis and, as much as possible, to utilize computing both for motivation and for problem working. The first parts of this paper deal with what has been done in the first semester, while the later parts deal with what is currently being done in the spring semester.

1. INTRODUCTION TO PROBABILITY AND STATISTICS THROUGH DISTRIBXTION-FREE METHODS

The first two weeks of the course were spent on an introduction to the binomial distribution with $p = 1/2$, well motivated with coin examples. The question, Is a coin really fair? led naturally to the notion of a hypothesis test, and, among other things, each student tested whether pennies balanced on edge have probability 1/2 of toppling heads up when the surface on which they are balanced is vibrated. (Conclusion: No. The probability is approximately 3/4, dependent somewhat upon the smoothness of the surface.)

The students were then introduced to the 2x2 chi-square test, the Wilcoxon test, and the Kendall test as three very useful statistical tools. They gained practice in using them by working some exercises derived from a questionnaire I gave at Northwestern University in 1969. They also gained a healthy respect for the amount of time required to apply a statistical test to a large amount of data. Distribution-free tests are fairly easy to explain in terms of discrete probabilities, so the students generally seemed intellectually satisfied. They were told that there would be much more on the limiting distribution of the test statistics later in the course, but for the time being they were safe in using chi-square with 1 df for all of them. (Note: I squared the usual Wilcoxon and Kendall large-sample statistics and furnished the students with a cumulative probability chi-square table of my own manufacture. Thus, there was no need to take square roots.)

I am convinced that the brief introduction to probability and statistics described above gave them considerable motivation for the rest of the course. To give them even more motivation, I required each student then to begin work on a research project of his own choosing. Each student was allowed to submit up to ten questions, answerable with 'yes', 'no', or an integer from 0 to 99, all to be typed on a common questionnaire. Each of the twenty students was then given five copies of the questionnaire and required to fill one out himself and have four friends do the same. Each student punched data cards from his own set of questionnaires, 3 cards per, for a total of fifteen cards. Yesses were punched as 1s and noes as 0s. I checked the cards with overlay cards to make certain the numerical answers were in their proper columns, found and had the students correct a half-dozen mispunches, and then reproduced the 300 card data deck for each student. (A disk file was considered, but I decided multiple decks would be more flexible, especially for those students not familiar with programming.)

My requirements for the research project were simple: Run at least one of each test, chi-square, Wilcoxon, and Kendall, and find at least two relations significant at the .05 level. Two students indicated that they were afraid they would not be able to learn to use the computer in time to complete their projects, so I furnished them with listings of the data deck. Ultimately, however, they did run their tests on the computer, as did everyone else. I furnished everyone with FORTRAN subroutines for the three tests, so the primary concern of each student was to learn enough FORTRAN to do input and, if necessary, to combine yes-no answers into numerical scales. This is backwards from the way computing should be taught, but it was effective. Most students examined the subroutine listings out of natural curiosity to see why they worked. An example of a main program and subroutine is given below.

As I expected, no student experienced great difficulties following the guidelines to a successful run. Debugging was done with one-tenth of the full data deck, and each student was expected to compute these small-sample answers by hand, to make sure he agreed with the printed

```
C FIRST TWO ANSWERS ARE THE YES-NO ANSWERS
      COMMON IARRAY(4,100), I, J, N
      N = 100
      READ (5,50) IARRAY
   50 FORMAT (5X, I1, /, 7X, I1, 3X, I1, /, 10X, I2)
      WRITE (6,60) IARRAY
   60 FORMAT (4I6)
      I = 1
      J = 3
      CALL WILCXN
      I = 2
      J = 4
      CALL WILCXN
      STOP
      END

      SUBROUTINE WILCXN
      COMMON IARRAY(4,100), I, J, N
      M = 0
      DO 1 K = 1,N
      IF (IARRAY(I,K) .EQ. 0) M = M + 1
    1 CONTINUE
      U = 0.0
      DO 2 K = 1,N
      IF (IARRAY(I,K) .EQ. 1) GO TO 2
      DO 3 L = 1,N
      IF (IARRAY(I,L) .EQ. 0) GO TO 3
      IF (IARRAY(J,L) - IARRAY(J,K)) 3,4,5
    4 U = U + 0.5
      GO TO 3
    5 U = U + 1.0
    3 CONTINUE
    2 CONTINUE
      PROD = M*(N-M)
      V = U - 0.5*PROD
      W = 12.0*V**2/(PROD*FLOAT(N+1))
      ISIGN = V/(ABS(V) - 1.0E-20)
      WRITE (6,60) I,J, M,N,ISIGN, U,V,W
   60 FORMAT (2I7, 10X, 3I7, 10X, 3F20.7)
      RETURN
      END
```

output. Debugging and short runs were in WATFIV on the university's IBM 360/50. One could expect to run 100 chi-square, 6 Wilcoxon, or 2 to 3 Kendall tests per minute of WATFIV time, varying considerably, of course, with the data. Five or so students expressed an interest in running large numbers of Wilcoxon and Kendall tests, so I let them go on to run in our regular FORTRAN G, after making certain they understood the philosophy of hypothesis testing. Total cost of the computer time was approximately fourteen dollars per student, very reasonable in my opinion.

2. THE DEVELOPMENT OF ONE-DIMENSIONAL PROBABILISTIC MODELS

After the five week introduction to probability and statistical hypothesis testing, we returned to begin a deeper study of the subject. I developed the notion of a random variable, discussed the two important classes -- discrete and continuous -- and then covered at some length the following examples of random variables: binomial, Poisson, negative binomial, hypergeometric, normal, lognormal, gamma, and the two important special cases of gamma, exponential and chi-square.

Using CPS BASIC, I did simulations of some of these random variables. Regrettably, the department did not have sufficient funds for me to make photocopy handouts of these to give my students. Only later did I realize that I could have inserted a spirit master in the typewriter terminal and had both the program listing and the output printed out ready to be duplicated cheaply. I had my students repeat my coin flipping simulation with real coins to see that they would obtain essentially the same results. They were required to shake 20 pennies together and then spill them out, counting how many heads they obtained. After 100 repetitions, they graphed their results on normal by linear paper, to observe how close the points came to falling on a straight line.

More interesting than simulations, however, were studies of the data from their questionnaires. For instance, I plotted heights of the male students on normal by linear paper and obtained a near straight line scatter. Then I plotted parents' incomes on normal by linear paper, obtaining a very pronounced convex-downward scatter. Finally, I replotted the incomes on normal by logarithmic paper, which caused them to straighten out dramatically. The students were given the assignment of plotting on normal by linear paper numerical scores of their own choosing from the questionnaire data and interpreting their results. From their work they have acquired some appreciation of the fact that certain kinds of data are very close to normally distributed while others are very wildly non-normal.

The remainder of the first semester, approximately two weeks, was spent studying the behavior of sums and other combinations of independent random variables. The second semester will continue on that topic, in particular discussing in some detail but not proving a fairly strong central limit theorem.

3. STATISTICS WITH THE AID OF COMPUTING

The second semester of the course is intended to be a very practical one. I know of no better way to teach practical statistics than to do three things simultaneously:

a) Cover the important techniques in as much depth as the students' backgrounds permit.

b) Let the students develop algorithms for the simpler kinds of tests.

c) Let the students simulate wildly non-normal random variables and see for themselves how much the nominal Type I and Type II errors of the parametric tests change.

In the first two weeks, the material covered will still be preparatory, up to the central limit theorem. One simple assignment will be for the students to use the random number generator in CPS BASIC to verify the central limit theorem. I plan to have them sum RND by itself and also to sum RND , to see the difference in speed with which the central limit result becomes a useful approximation. Their results will once again be graphed on normal by linear paper.

When I teach normal statistics, I make it clear that the central limit theorem does guarantee that for any nice population distribution there exist sufficiently large sample sizes such that the Type I error becomes as close as one desires to the nominal level. The problem, then, is deciding what sample sizes are sufficiently large for non-normal populations occurring in practice. Using CPS BASIC, we can quickly run simulations of the t test on different-shaped populations to see what sample sizes are sufficiently large.

I expect to cover normal statistics through to the simpler forms of analysis of variance, and expect to give the students homework problems that they will work on the CPS terminals. They will work with the same data from the questionnaires, this time held on disk file. The BASIC language is so easy to learn that they should be able to work their way up through normal techniques, from a simple test for the equality of a mean to a fixed number all the way to analysis of variance. My experience on CPS is that the execution speed is not adequate for the large numbers of counts in large-sample Wilcoxon and Kendall tests, but it is fast enough for the arithmetic needed in parametric tests.

I expect to cover some other distribution-free tests, and then compare powers of various tests in the face of skewness and too full tails. Again, I will have the students run simulations on some wild populations. This topic leads naturally to the topic of censoring data and to the topic of jackknifing. I have not given much thought as of this writing (January 1971) how the computer will enter the topic of censoring data, if it will at all. However, the computer is the natural tool to use with the jackknife, since it makes the repetitive calculations required possible to do.

4. STUDENT ENTHUSIASM

There is no doubt at all that integrating computing into a probability and statistics course is very beneficial to the students. I mention again that my students have had little or no computing experience before encountering computing in my course. The fact that high speed computing removes the tedious, painstaking calculations from statistics means that there is much more time for the student to concentrate on the essentials of the subject. At the same time, he is very well motivated to begin learning to program, because he sees before him a real, flesh-and-blood application of the computer. This is especially important in learning input-output techniques, the stumbling block in most programming courses. In picking data from the cards to analyze, he sees the necessity of the FORMAT statement very clearly.

There is, of course, a certain amount of shock to the student when he learns what is expected of him in his research project. To lessen this, I made it clear that it would be possible to perform the tests by hand, and that anyone who so wished could obtain from me a listing of the data deck to make that task easier. It turned out, however, that everyone succeeded in making use of the computer. Next year that first encounter with the machine should be easier, because I had one student write up a handy guide to setting up a main program. Another student--somewhat experienced in computing--wrote up an explanation of the three sub-routines, complete with flowcharts. Those students who volunteered their feelings at the end of the semester indicated that until they had obtained their first successful run, they were worried, but after they saw how easy it was they found the work on the projects quite enjoyable.

Such a large questionnaire as a class generates is a gold mine of information which can be used for a long while to come. It has a personal flavor, since some questions from each student are on it, so examples taken from it are very likely to be well received. I have so far had great success using it in connection with the probabilistic models portion of the course, and I intend to use it as much as possible in the second semester. In planning homework involving the data, I have found it helpful to have some feel for the variability of the numerical data. Accordingly, I have already obtained a listing of all numerical data items, in order.

In summary, my students have learned approximately twice as much information on the whole and many times more information that is practical than they would have learned in the traditional course. The computer takes the drudge out of statistics and it also encourages them to keep on learning outside of class lectures. They are forced to work harder, but in general they are thankful for the opportunity.

5. COMPUTER SYSTEMS

It is fortunate for my course that the university has a flexible setup, both batch process and time-sharing, because each system has its advantages at different times in the course. If I had a choice of only one system, however, I would be in favor of batch process, because the distribution-free tests at the beginning of the course would require far too much central processor time in time sharing. Batch process would not give such good feedback on the short homework problems later in the course, but at least it would be feasible.

I believe, then, that anyone whose installation is limited to batch mode could just as well achieve an effective use of the computer in a similar statistics course.

COMPUTING IN PROBABILITY OF COURSE

J. Laurie Snell
Dartmouth College
Hanover, New Hampshire

Probability and computing have long been friendly and it is natural to continue this friendship using the modern computer. Professor Kemeny and I did this in an experimental course this year. While the course was basically a first college course in probability, we had the advantage of being able to assume that all the students knew finite probability concepts and basic computing, both being taught in the first two years mathematics program at Dartmouth. With our time sharing system they had easy access to the computer.

We did not use a text and made computer assignments given once a week a significant part of the student's work outside the classroom. We started the course with a discussion of fluctuation theory in coin tossing. The first computer assignment tested whether the students would discover the highly unintuitive arc sine law for the number of times in the lead in a penny matching game. We considered also the last time of equalization of fortunes. This assignment is shown in Figure 1 together with a typical answer.

The students saw immediately that in twenty matches the numbers near the expected value of 10 are less likely than the extreme values of 0 or 20, i.e. never or always in the lead. Most also guessed that the distribution for the times in the lead and the last time the fortunes were equal were the same. Having guessed these results, we proved that in 2n matches

$$\Pr\left[F(k) = 2k\right] = \Pr\left[G(k) = 2k\right] = \frac{1}{2^{2n}} \binom{2k}{k} \binom{2n-2k}{n-k}$$

using Feller's very elegant treatment based on the reflection principle. Again using the computer and Stirling's formula, the arc sine limit theorem was illustrated. We asked the class to modify their programs to observe the number of times in the lead, counting only the experiments where the fortunes were equalized at 20 trials. Here we hoped that they would discover the surprising result that now all the possibilities for the number of times in the lead have the same probability, a result which was first suggested by computer experiments. However, the students complained that they could not find any distribution in the case since all possibilities seemed to have about the same probability!

We tried in this example and in other situations to show how one very simple program needed could be adapted to a variety of problems. This worked particularly well for the following three problems. The first related to a problem in judging extrasensory perception experiments. A subject is given a sequence of experiments in which he is to determine on each trial whether a card turned face down has a circle or a square. If he guesses, how much advantage does he have when allowed optional stopping and what is the optimal stopping rule to maximize the expected proportion of correct responses?

The optimum rule is indicated in Figure 2. From this the students noticed that they should continue to play as long as the number correct was less than a number that depends on the number of the experiment. They were encouraged to prove this fact. The obvious conjecture that one should stop as soon as he has more than 1/2 correct was shown to be false by observing, for example, that after 14 trials the optimal rule is to continue if 9 answers have been correct. A slight modification of the program showed that the value of ESP score of .771 achieved by guessing and optimal stopping was reduced to only slightly better than .5 if the subject could only stop during the second half of the experiment.

The second problem is the famous statistical problem of discriminating between two probability models, both independent trials, one with probability p_0 for successes and the other with probability p_1 for success. First using the decision theory approach we considered the example of an a priori probability .75 for the hypothesis H_0 that p_0 was correct. We assumed a cost of \$1 per experiment, and a loss of \$20 for accepting p_0 when p_1 is correct and \$10 for accepting p_1 when p_0 is correct. We showed that again the problem was a simple optimal stopping problem. The result of the computation of the optimal rule is shown in Figure 3. The student was encouraged to try different values of the parameters and to print out different kinds of information. For example in Figures 4 and 5 the updated probabilities for the hypothesis H_0 that p_0 is correct are printed out at the points where the testing continues. It is observed that for two different initial probabilities of H_0, essentially the same rule is obtained. Further from observing these values the rule to continue as long as $A < Pr[H_0 | j \text{ successes}] < B$ for suitable A and B is strongly suggested. When the probability falls below A we stop and accept the hypothesis that p_1 is correct. If it goes above B we stop and accept H_0. This led naturally to Wald's famous sequential ratio test and a discussion of methods of describing this test for given error probabilities. Simulation was then used to show that the test was effective in discrimination.

We used this second example to show how one might have been led to a famous mathematical result by the flexibility of computer programs. The last of the three stopping examples was used to illustrate the discovery of a mathematical result which has yet to be adequately explained.

The problem starts with a simplified model for stock increases. It is first assumed that a stock rises one unit or falls one unit in price each day with probability 1/2. In this case optimal buying or selling represents a fair game and has value 0. We then showed how to modify the probability model if we have a prediction concerning the rise or decline in the stock after 10 days. The standard optimal stopping procedure was applied and two results were observed. First even in the face of a prediction that the expected increase after 10 days was negative the value under optimal stopping was now positive. Even negative knowledge has more value than no knowledge! Secondly, when two predictors predicted distribution with mean 0 after 10 days, the optimal strategy is to sell when the price is low if the variance is greater than 10, and exactly the opposite, sell when high when this variance is less than 10. See Figures 6 and 7. This surprising result has yet to be explained mathematically.

The three problems just discussed are attractive because the same general mathematical theory applies to all of them and because it is a simple matter to write a general purpose program which requires only modest changes to go from one situation to the other.

Of course other topics such as queueing theory and branching theory were discussed in the course. The computer played again a central role in the understanding. As a last example I mention the comparison of the law of large numbers for coin tossing and the basic limit theorem for branching processes. In coin tossing if S_n is the number of heads which occur in n tosses the students are quite prepared to believe that S_n/n will approach 1/2 under any simulation experiment. There is little point to asking them to do this. However, consider the population problem with each object producing a random number of new objects with mean μ. Then if Z_n is the population after time n the basic result now is that the population grows geometrically and Z_n/μ^n tends to a limit. However, in contrast to the coin tossing experiment the limiting value will vary when different simulations are carried out reflecting the fact that Z_n/μ^n tends to a limiting random variable and not a constant. Simulation now plays a very convincing role.

To summarize I believe that as a supplementary tool to understanding probability the computer is of enormous benefit. I also believe that in time the students will learn to discover significant new results in probability because of the ease with which they can modify existing probability models amenable to mathematical treatment.

FIRST ASSIGNMENT: WRITE A PROGRAM TO SIMULATE A GAME OF PENNY MATCHING BETWEEN PETER AND PAUL FOR TWENTY MATCHES. REPEAT THIS GAME 1000 TIMES AND LET F(K) = THE NUMBER OF TIMES THAT PETER LEADS EXACTLY K TIMES AND G(K) = THE NUMBER OF TIMES THAT THE LAST TIME THE FORTUNES ARE EQUAL IS AT TIME K.

```
RUN

ARCSINE

K               F(K)            G(K)

 0               181             180
 2               109             92
 4               73              83
 6               58              65
 8               61              53
 10              65              63
 12              67              65
 14              68              58
 16              73              64
 18              81              105
 20              164             172
```

FIGURE 1

```
OLD OPSTOP
READY

RUN

OPSTOP    21 APR 71  14:46

OPTIONAL STOPPING RULE FOR ESP EXPERIMENT WHEN THIRTY
TRIALS ARE ALLOWED.  STOP AT POINTS MARKED  .  AND
CONTINUE AT POINTS MARKED  *  .


VALUE = 0.771

*
* .
* * .
* * . .
* * * . .
* * * * . .
* * * * . . .
* * * * * . . .
* * * * * . . . .
* * * * * * . . . .
* * * * * * . . . . .
* * * * * * * . . . . .
* * * * * * * . . . . . .
* * * * * * * * . . . . . .
* * * * * * * * . . . . . . .
* * * * * * * * * . . . . . . .
* * * * * * * * * . . . . . . . .
* * * * * * * * * * . . . . . . . .
* * * * * * * * * * . . . . . . . . .
* * * * * * * * * * * . . . . . . . . .
* * * * * * * * * * * . . . . . . . . . .
* * * * * * * * * * * * . . . . . . . . . .
* * * * * * * * * * * * . . . . . . . . . . .
* * * * * * * * * * * * * . . . . . . . . . . .
* * * * * * * * * * * * * . . . . . . . . . . . .
* * * * * * * * * * * * * . . . . . . . . . . . . .
* * * * * * * * * * * * * * . . . . . . . . . . . . .
* * * * * * * * * * * * * * . . . . . . . . . . . . . .
* * * * * * * * * * * * * * * . . . . . . . . . . . . . .
* * * * * * * * * * * * * * * . . . . . . . . . . . . . . .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

TIME: 0.870 SEC.
READY
```

FIGURE 2

```
OLD BAYES
READY

RUN

BAYES     21 APR 71  14:34

OPTIMAL STRATEGY FOR TWENTY TRIALS  *  MEANS CONTINUE

W0 =  0.75      P0 =  0.75      P1 =  0.5      C =  0.1

VALUE = 2.68

*

*  *

*  *  *

*  *  *  *

.  *  *  *  .

.  *  *  *  *  .

.  .  *  *  *  *  .

.  .  .  *  *  *  .  .

.  .  .  *  *  *  *  .  .

.  .  .  .  *  *  *  *  .  .

.  .  .  .  .  *  *  *  .  .  .

.  .  .  .  .  *  *  *  *  .  .  .

.  .  .  .  .  .  *  *  *  .  .  .  .

.  .  .  .  .  .  .  *  *  *  .  .  .  .

.  .  .  .  .  .  .  *  *  *  .  .  .  .  .

.  .  .  .  .  .  .  .  *  *  *  .  .  .  .  .

.  .  .  .  .  .  .  .  .  *  *  *  .  .  .  .  .

.  .  .  .  .  .  .  .  .  .  *  *  .  .  .  .  .  .

.  .  .  .  .  .  .  .  .  .  .  *  *  .  .  .  .  .  .

.  .  .  .  .  .  .  .  .  .  .  .  *  .  .  .  .  .  .  .

.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

TIME: 1.049 SEC.
READY
```

FIGURE 3

```
2 PRINT "VALUE OF PROBABILITY OF H0 AT CONTINUE POINTS"
346 PRINT FNR(W(I,N));
RUN

BAYES      21 APR 71   14:39

VALUE OF PROBABILITY OF H0 AT CONTINUE POINTS

W0 =  0.75      P0 =  0.75      P1 =  0.5       C =  0.1

VALUE = 2.68

 0.75

 0.6  0.82

 0.43  0.69  0.87

 0.27  0.53  0.77  0.91

.   0.36  0.63  0.84 .

.   0.22  0.46  0.72  0.88 .

.  .   0.3  0.56  0.79  0.92 .

.  .  .   0.39  0.65  0.85 . .

.  .  .   0.24  0.49  0.74  0.9 . .

.  .  .  .   0.32  0.59  0.81  0.93 . .

.  .  .  .  .   0.42  0.68  0.86 . . .

.  .  .  .  .   0.26  0.52  0.76  0.91 . . .

.  .  .  .  .  .   0.35  0.62  0.83 . . . .

.  .  .  .  .  .  .   0.44  0.71  0.88 . . . .

.  .  .  .  .  .  .   0.29  0.55  0.78 . . . . .

.  .  .  .  .  .  .  .   0.38  0.64  0.84 . . . . .

.  .  .  .  .  .  .  .  .   0.47  0.73  0.89 . . . . .

.  .  .  .  .  .  .  .  .  .   0.57  0.8 . . . . . .

.  .  .  .  .  .  .  .  .  .  .   0.67  0.86 . . . . . .

.  .  .  .  .  .  .  .  .  .  .  .   0.75 . . . .  .  .  .

.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

TIME: 1.022 SEC.
READY
```

FIGURE 4

```
90 DATA .5
RUN

BAYES      21 APR 71  14:41

VALUE OF PROBABILITY OF H0 AT CONTINUE POINTS

W0 =  0.5       P0 =  0.75      P1 =  0.5       C =  0.1

VALUE = 2.94

 0.5

 0.33  0.6

 0.2  0.43  0.69

.   0.27  0.53  0.77

.  .   0.36  0.63  0.84

.  .   0.22  0.46  0.72  0.88

.  .  .   0.3  0.56  0.79  0.92

.  .  .  .   0.39  0.65  0.85  .

.  .  .  .   0.24  0.49  0.74  0.9  .

.  .  .  .  .   0.32  0.59  0.81  0.93  .

.  .  .  .  .  .   0.42  0.68  0.86  .  .

.  .  .  .  .  .   0.26  0.52  0.76  0.91  .  .

.  .  .  .  .  .  .   0.35  0.62  0.83  .  .  .

.  .  .  .  .  .  .  .   0.44  0.71  0.88  .  .  .

.  .  .  .  .  .  .  .   0.29  0.55  0.78  .  .  .  .

.  .  .  .  .  .  .  .  .   0.38  0.64  0.84  .  .  .  .

.  .  .  .  .  .  .  .  .  .   0.47  0.73  0.89  .  .  .  .

.  .  .  .  .  .  .  .  .  .  .   0.57  0.8  .  .  .  .  .  .

.  .  .  .  .  .  .  .  .  .  .  .   0.67  0.86  .  .  .  .  .

.  .  .  .  .  .  .  .  .  .  .  .  .   0.75  .  .  .  .  .  .

.  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .

TIME: 1.195 SEC.
READY
```

FIGURE 5

```
RUN
STOCKS    21 APR 71  15:18

OPTIMAL SELLING STRATEGY FOR THE STOCK MARKET PROBLEM
IF FOR THE END OF A TEN DAY PERIOD THE PREDICTOR
PREDICTS THAT THE PRICE OF THE STOCK WILL HAVE
EXPECTED INCREASE OF 0 AND VARIANCE OF 8
* MEANS HOLD THE STOCK AND  .  MEANS SELL

VALUE = 0.152

*

*    *

*    *    .

*    *    *    .

*    *    *    .    .

*    *    *    .    .    .

*    *    *    *    .    .    .

*    *    *    *    .    .    .    .

*    *    *    *    *    .    .    .    .

*    *    *    *    *    .    .    .    .    .

.    .    .    .    .    .    .    .    .    .    .

TIME: 0.424 SEC.
READY
```

FIGURE 6

```
RUN

STOCKS     21 APR 71   15:26

OPTIMAL SELLING STRATEGY FOR THE STOCK MARKET PROBLEM
IF FOR THE END OF A TEN DAY PERIOD THE PREDICTOR
PREDICTS THAT THE PRICE OF THE STOCK WILL HAVE
EXPECTED INCREASE OF 0 AND VARIANCE OF 12
* MEANS HOLD THE STOCK AND  .  MEANS SELL

VALUE = 0.14

*

*    *

.    *    *

.    *    *    *

.    .    *    *    *

.    .    .    *    *    *

.    .    .    *    *    *    *

.    .    .    .    *    *    *    *

.    .    .    .    *    *    *    *    *

.    .    .    .    .    *    *    *    *    *

.    .    .    .    .    .    .    .    .    .    .

TIME: 0.417 SEC.
READY
```

FIGURE 7

THE COMPUTER AND FINITE MATHEMATICS

Rev. J. Richard VandeVelde, S.J.
Loyola University
Chicago, Illinois

I. Introduction

In the last few years the digital computer has had its impact on the undergraduate curriculum in mathematics, notably in the areas of the calculus, differential equations, and numerical methods. The most obvious example of this is the CRICISAM text in calculus which is computer-oriented. This development is not at all surprising since the digital computer has tremendous "number crunching" potential. However, the computer also has inherent symbol manipulating capabilities, and of course, a powerful logical structure. It is these characteristics rather than its numerical strengths which we have attempted to exploit in the course in Finite Mathematics.

For the past two and a half years those involved in teaching the course in Finite Mathematics have evolved a package of programs to be used in conjunction with this course. Not all of the programs are used all of the time, but they form a pool from which to draw.

Finite Mathematics is a somewhat ill-defined term. The content of a course in Finite Mathematics varies from school to school as well as from year to year even in the same school. We can take the following as a fairly representative list of topics which are usually treated in a course in Finite Mathematics: propositions and truth tables, logical arguments, set theory, relations, functions, vectors and matrices, linear equations, probability, Markov chains, inequalities, linear programming, and theory of games.

The audience for a Finite Mathematics course at Loyola is drawn from liberal arts majors, generally non-mathematically oriented students, who are taking the course to fulfill a mathematics requirement. The major fields of interest of the students enrolled in the course include psychology, sociology, political science, anthropology, and to some extent, biology and the humanities.

In view of the audience and the subject matter, we decided to emphasize the role of the mathematical model in various fields. Thus we are constantly stressing that we are building a model of some situation, structure, or set of relationships. The propositional calculus with logical implication models the way we formulate arguments; relations and graphs can mirror the structure of a small society, a network to be traversed, etc. Probability and statistics, especially Markov chains, present us with models of behavior which are not deterministic, yet correspond to some of the experiences of the students. In all the models it is important to delimit both the aspects being modeled, and those being ignored.

II. Computing Environment

The computing facilities which have been available to us and the students have changed considerably over the last three years. In spite of these changes, we have found that the goals of the course could be substantially achieved in any one of the environments. Initially, Loyola University was a member of the Illinois Institute of Technology Regional Computer Network. In practice this meant that all of our computing on the undergraduate level was done 1) via teletype over the phone lines, 2) in batch mode, and 3) processed by IITRAN or FORTRAN. IITRAN was implemented on an IBM 360/40 as a single pass load and go compiler. It was constructed to provide rapid processing of student or one-time programs. Later we obtained a timesharing service, still with teletype input, using BASIC. At present our undergraduate computing is concentrated on one of Loyola's own IBM 360/40's with input and output being via a 2780 remote terminal. The languages currently in use are FORTRAN and PL/1, with both standard FORTRAN IV and a fast FORTRAN compiler, WATFIV, being available.

We have thus used both batch and timesharing modes, as well as punched tape and card input. Apart from certain minor differences the results have been substantially the same.

III. Topics Covered

The following is a list of some of the topics covered and how the computer was used in conjunction with them. This list should not be construed as an outline of the course, since, as indicated above, not all of the topics are covered each time the course is taught.

Random Numbers. One of the assignments given in the course was that of obtaining a table of random numbers. This was done for several reasons. First of all, such a table would be used throughout the course, especially in the portions of the course dealing with probability, game theory, or simulation. Secondly, the amount of work the student had to do to obtain his own table of random numbers was minimal -- five punched cards. Each student was given a handout indicating just what format the cards had to have. In addition, each student supplied on one of the cards his own seed for the random number generator, insuring that he would have a unique table of numbers. Appendix A contains an example of the output for this program. The directions given the student resulted in his obtaining two copies of the table. In this way he could turn one in as an assignment and keep the other one for his use.

The approach here is admittedly one of a black box. There was no real attempt to explain how the numbers were generated, nor was there any detailed explanation of what the strange characters and words on the punched cards meant.

The algorithm used to generate the random numbers was obtained from the Communications of the Association for Computing Machinery, though any suitable generator would do. It was modified to give a tally of the number of occurrences of each digit in the 1000 digits in the table. The program is in FORTRAN.

Queueing Simulation. A simple example of customers entering a supermarket at random intervals and taking different amounts of time to shop is discussed in the class. The basic problem is that the customers must queue up for checking out at the only checkout counter. The interarrival times of the customers and the length of time which they shop are determined by two random variables. The students determine these by using their tables of random numbers as arguments. This example can be worked by hand. After this the students run the same problem on the computers, needing only to supply a seed for the random numbers. The printout is a table of arrival times and other pertinent statistics, such as the average wait per customer. The computer thus is used to do essentially the same thing which the student can do, but the element of saving of time, as well as replication of the same simulation are two aspects which are pointed out. This computer program also serves to broaden the students' conception of just what a computer is good for. Program has run in BASIC and FORTRAN.

Logic Programs. If the course in Finite Mathematics treats of the propositional calculus and valid argumentsc then the computer can be used to perform automatically many of the same types of work which the students usually do. Filling out a complete truth table for any reasonably complicated proposition is both tedious and subject to error. We have two computer programs which take some of the drudgery out of this process, but do not supply for the knowledge of the student. The computer is doing what the student could do. In this sense the computer is not totally a black box. The two programs we used are called SYMBOLOG and VALIDITY. Both were implemented nicely and easily in IITRAN and with little difficulty in PL/1. FORTRAN does not seem to be as suitable because of the lack of symbols for the logical connectives in FORTRAN.

SYMBOLOG takes as input an integer, N, denoting the number of atomic propositions P, Q, R, S, . . . out of which the complex proposition is built. Also the proposition itself must be fed to the computer. The output is either a) the entire truth table, or b) only those lines when the proposition is true. This option is specified by another input parameter.

VALIDITY takes as input a sequence of N propositions ($0 \leq N \leq 10$), considered as premises to the argument, and another proposition considered as the conclusion. In addition, the student must indicate the number, M, of atomic propositions which appear in his premises. The output is either a) a statement to the effect that the argument is valid, or b) a list of the cases in which it is invalid, i.e. the truth values for the atomic propositions which make all the premises true and the conclusion false. One should note that if the number of premises is 0 then one is effectively asking for the cases when the "conclusion" is false independently of any premises. In such a case, the validity of the argument is equivalent to the fact that the "conclusion" is a tautology. The input in PL/1 would look like:

```
2       4
PREM(1) = (P & Q)    <= S
PREM(2) =  R | (P & S)
CONCL = R = (¬P & Q)
```

where |, &, <=, =,¬ represent respectively 'or', 'and', 'if...then', 'if and only if', and 'not'. It is a bit unfortunate that " <=" looks like an arrow pointing to the left, especially since one of the standard symbols for if...then is "→" which is an arrow pointing to the right.

Again, I would like to emphasize that as far as the student is concerned this is really not a black box, perhaps only a gray box, in that most students are willing to believe that the computer can go through the same steps which they would have to in order to determine the validity of the argument -- and they are right.

Sets, Relations, Graphs. For this section of the course we have developed several computer programs to help solidify the concepts presented, or to work programs on a more realistic scale or size. Set theory is usually introduced as a tool, a style of thought, a language, and not so much as an end in itself. The language of relations, as subsets of Cartesian products provides a vehicle for discussing a number of interesting problems.

We have found that relations are familiar to the student, and he enjoys discussing common ordinary relationships from a technical point of view. We introduced the notions of symmetry, reflexivity, and transitivity, and their variations. Once this is done one can discuss the various methods of representing or symbolizing the relationships. Two methods are useful: matrices and directed graphs. There are advantages to using different ways of representing or modeling the relations. Certain properties are more immediately obvious in one representation rather than the other. Students come to see that there is not necessarily a single best way to represent something.

The program RELATION examines a relation for reflexivity, symmetry, and transitivity. Such a program is not difficult to write, in fact our program has been run successfully in BASIC, FORTRAN, and in PL/1. In addition to testing the above properties, the program tests for irreflexivity, asymmetry, and in case one or the other property fails, the program also indicates at least one pair of "individuals" for which it fails, or in the case of transitivity it yields a triple (I,J,K) such that I is related to J, J is related to K, but I is not related to K. Sample output to RELATION is given in Appendix B.

Another program we have is PATH which finds the length of the shortest path between any two points in a network or directed graph. The input to this program is a matrix of the direct distances between pairs of points. As presently written the program only gives the length of the path, not the actual nodes along this path -- however, one can leave something to the student.

Finally there is the program CIRCUIT which searches and finds all the elementary circuits in a network. An elementary circuit is a path containing each vertex at most once in which the first and last vertices of the path are the same.

The above programs are not used until the ideas and reasons for using directed graphs or matrices have been introduced by concrete examples. Again, the computer here is used mainly as a labor-saving device not as a magic box giving answers which the student could not hope to achieve himself. Sample output to CIRCUIT is given in Appendix C.

Vectors and matrices. Having been introduced to matrices in an almost non-mathematical setting, the student is usually ready to grant their usefulness. The "ordinary" use is for systems of equations. If simultaneous equations of equations are to be treated in the course, a program which illustrates the Gaussian elimination process can be used. The one we have used shows all the intermediate stages of the process so that the student can follow the process without having to do all the messy divisions and multiplications.

Probability. In our treatment of probability we have under-played the comb'natorial sides of the subject. It seems difficult to get students interested in the number of ways you can seat 14 couples at a round table such that no one is next to his spouse, or in the number of ways in which you can fly so many flags on a pole. The ideas involved in Markov chains, however, seem more interesting to the student. Again, matrices and vectors provide the necessary vehicle, and lend themselves to this model of behavior.

MARKOV is a program which calculates the probability distribution after n stages given an initial probability vector and the transition matrix. If one uses a language which has matrix handling capabilities built in -- as some versions of BASIC do -- this is almost a trivial programming exercise. However, it can be done in any languages which have arrays. We made use of this capab'lity to help exemplify the notion of a fixed point for a stochastic process. The student merely had to input the transition matrix, and the initial probability vector plus the number of stages he wishes the process to go through. To avoid excessive printout, or infinite loops in the case of a non-converging process, we built in a limiting factor of 30 iterations. Most processes settled down well before that, if they were going to converge at all.

This is one program which worked best, from a learning point of view; when it was used in a conversational environment. The program was so written that after the fixed point had been reached the student was queried as to whether a) he wished to try a different initial starting vector with the same matrix, b) try a new matrix and new start, or c) stop. As written in FORTRAN, this flexibility is reduced, and the student must decide before he runs the program whether he wishes different starting vectors and what they are. Of course he can run it once, and then submit a second job, etc.

Students were frequently impressed by the speed at which a Markov process converged to its fixed point, even for widely differing initial vectors.

Game Theory. If the topic of Game Theory is covered in a course in Finite Mathematics, the student has another opportunity to use his table of random numbers. In addition to this, a simple program can be written to solve any 2 by 2 matrix game. The program which we have used does not solve a given game, but rather plays it repeatedly according to row and column strategies determined by the user. GAME takes as input the matrix of the game, and the desired row and column strategies, together with the number of times which the game is to be played. The output consists of a display of the matrix, the first few plays of the game as determined by the desired strategies, and finally the actual strategies played together with the average value per game. This program provides an empirical "proof" that the value of the game as determined by the mathematical theory is actually what happens in the long run. (A similar idea can be exploited in a probability and statistics course to illustrate the Central Limit Theorem.) See Appendix D for sample output to GAME.

Linear Programming. Any treatment of linear programming which attempts to introduce the simplex method of solution always has the disadvantage that the more realistic the problem the more tedious and error-prone become the calculations. LINPRO is a FORTRAN program which we have used to solve a linear programming problem. It has the virtue that it is a straightforward application of the simplex method and, if the user desires, all the intermediate tableaux are also displayed. In this way, the student can follow all the calculations without actually having to perform them.

Other topics. There are several other programs available to the teachers of Finite Mathematics at Loyola which had their origins in other courses, but have on occasion been used in a Finite Mathematics course.

CURVEFIT is a program which fits, by the method of least squares, 6 different types of curves to the data input by the user. As implemented in BASIC the user could determine which of the six types he wished. In its FORTRAN implementation all six types are given automatically. In both versions a coefficient for the goodness of fit is given.

TABLE is a simple program which lists the values of one, two or three functions over some range of the independent variable with a specified increment. In Calculus courses this is useful in the study of parametric equations.

SENTENCE is a program which realizes a small generative grammar. The output is a string of sentences formed in accordance with the rules for their generation. It has a built-in random number generator which is used to determine certain options which occur as the program progresses. The vocabulary supplied is quite limited, but the sentences generated are syntactically correct. None of the teachers of Finite Mathematics have ever dwelt on this topic but it has been mentioned as a model of English grammar and illustrates a more humanistic use of the computer to test whether the grammar is an adequate model of English.

SOLITARE is a computer program which actually plays the standard solitaire game of Klondike. The input is merely a seed for the random number generator to shuffle the cards. The output consists of a display of the array of cards as they are dealt out, together with a statement of the moves made by the "player" as he goes through the game. When no more plays are pos0ible the computer then prints out the array of cards as they are at the end of the game together with the statistics on how many cards the "player" was able to move to the top four piles (the foundations).

Question of Programming. One serious question which we have had to ask ourselves is whether we would actually teach some computer programming in the course itself. Generally the answer has been 'no'. However, this past year, with the arrival of our WATFIV compiler with the free-form input/output one of the major obstacles to FORTRAN programming for the beginner was removed. The textbook used for Finite Mathematics this last year also had a chapter on the computer. Hence, this year we introduced the computer and actually taught a very stripped-down version of FORTRAN, e.g. no DO-loops, no subroutines.

It is interesting to note that very much can be done with the so-called Ten-Statement FORTRAN, provided you have some method of free-form input and output. The ten statements are:

```
     REAL                     INTEGER
     READ,                    PRINT,
     IF (logical) GO TO n     GO TO n
   n CONTINUE                 Variable = expression
     STOP (or CALL EXIT)      END
```

In this subset of FORTRAN the only numbered statements are the dummy CONTINUE statements, and the IF statement has always a GO TO n conclusion. Together with the ten statements one need only introduce the arithmetic operators and the logical connectives and fairly complicated programs can be written.

Conclusions. The students we have encountered in Finite Mathematics were not always pass6onately interested in the traditional applications of mathematics. However, when we could show them non-trivial applications of mathematics to other fields many of them admitted that they had never realized the breadth and utility of the mathematical model.

Although there are always students who are unaccustomed to the precision required (e.g. starting FORTRAN statements in column 7), most of the students found the experience interesting and not excessively time-consuming. One of the reasons for this was the emphasis on the black (or gray) box rather than having the student program most of the examples.

It is essential to have good handouts available so that class time is not consumed in describing the use of the teletypewriter or the keypunch. Likewise, a handout describing the input for a canned program is preferable to writing the same on the chalkboard. We are convinced that most of what we have done could be done in either a time-sharing environment or in a batch-processing one. We were aided immeasurably by having the WATFIV compiler with its almost instant turn-around time. Student jobs also received the highest priority in the job queue.

The use of the keypunch seems to have several advantages over the paper tape used in a teletype. One of these is the ease with which corrections are made, especially if not discovered until later. However, a timesharing system with a good editing facility would eliminate some of this difficulty. One student remarked that it was something of a status symbol to walk around the campus with the deck of computer cards and pages of printout.

Our experience at Loyola has been that the introduction of the computer by ways of examples, simple programs which the student can run, and the actual programming by the student has greatly increased the interest in the course without sacrificing too much time or content. The computer program assisted in consolidating the classroom materials and providing more detailed or lengthy examples of materials which would otherwise have taken too long to do in class. Finally, it made most of the students aware of uses of the computer they had never dreamed of. Perhaps some of them will be less prone to fold, spindle, or mutilate the next data processing card they receive in the mail.

APPENDIX A

```
      A TABLE OF RANDOM NUMBERS

SEED = 12355

13882   46650   22748   08603   48889   87832   60215   53354   99825   45162
35720   34164   77919   29440   26383   21002   81042   30858   43995   55998
11122   77213   87635   95823   68342   20627   47323   62706   32509   00179
33060   55727   31815   17660   46845   75897   67160   49209   85669   11905
00453   98886   41531   83052   79894   74214   23441   71157   74273   76167
32300   87390   96702   04899   66397   28672   34287   59640   18343   86902
08793   20459   95437   70281   93956   72279   00560   81598   16958   41183
30630   18953   50608   91018   87850   71476   10396   68091   50017   51910
07023   51013   50323   68400   10808   71082   86675   90949   59432   17191
28971   69527   15594   79247   97312   25253   97414   68442   93792   27917
05363   83686   14220   45639   30360   24868   52796   90390   81306   82198
27203   71190   69490   66477   18864   68038   73523   77884   38476   92924
04603   14249   79116   32859   50823   67653   39504   09732   24080   57105
25541   02753   23396   54696   38326   12823   59641   86235   77140   58931
92934   35812   33012   20088   71375   11448   15922   19183   66765   13112
24881   24326   88283   41825   59888   65608   26759   96686   00824   23938
90274   67485   87918   17217   81837   64223   92041   28524   08439   88110
22111   55980   42189   38044   79331   19493   02877   05027   42598   98946
99514   98049   42804   05436   02389   18018   78159   37975   41113   54127
10452   86553   07075   26273   99893   62289   89995   05478   85273   64953

TALLY

DIGIT           0      1      2      3      4      5      6      7      8      9
OCCURENCES    103    103    111    102     88     90     84     92    118    109
```

APPENDIX B

```
BELOW IS THE MATRIX OF THE RELATION.  A '1' IN THE I-TH ROW AND THE
J-TH COLUMN INDICATES THAT  I  IS RELATED TO  J .  A '0' IN THAT POSITION
INDICATES THAT  I  IS NOT RELATED TO  J .

      1  2  3  4  5
  1:  1  0  1  0  1
  2:  0  1  1  1  0
  3:  1  1  1  1  1
  4:  0  0  0  0  0
  5:  0  1  1  0  0

THE RELATION IS NOT REFLEXIVE.
FOR EXAMPLE, WE HAVE THAT  1 IS RELATED TO ITSELF,
WHEREAS  4 IS NOT RELATED TO ITSELF.

THE RELATION IS NOT SYMMETRIC.
FOR EXAMPLE, WE HAVE THAT  1 IS RELATED TO  5,
BUT NOT VICE VERSA.

THE RELATION IS NOT TRANSITIVE.
FOR EXAMPLE, WE HAVE  1 RELATED TO  3, AND  3 RELATED TO  2,
BUT 1 IS NOT RELATED TO  2 .
```

APPENDIX C

```
MATRIX OF CONNECTIONS

1:   2  5  0  0  0
2:   2  3  5  0  0
3:   5  0  0  0  0
4:   3  2  0  0  0
5:   1  4  0  0  0

CIRCUITS FOUND :

1  2  3  5
1  2  5
1  5
2  3  5  4
2  5  4
2
3  5  4
```

APPENDIX D

```
MATRIX OF THE GAME
     1          2          3          4          5          6          7
 |----------|----------|----------|----------|----------|----------|----------|
 |          |          |          |          |          |          |          |
1|   8.00   |   3.00   |   1.00   |   3.00   |   2.00   |   5.00   |   3.00   |
 |          |          |          |          |          |          |          |
 |----------|----------|----------|----------|----------|----------|----------|
 |          |          |          |          |          |          |          |
2|   6.00   |   6.00   |   0.00   |   0.00   |   2.00   |   4.00   |   1.00   |
 |          |          |          |          |          |          |          |
 |----------|----------|----------|----------|----------|----------|----------|
 |          |          |          |          |          |          |          |
3|   0.00   |   3.00   |   5.00   |   6.00   |   6.00   |   7.00   |    6.00  |
 |          |          |          |          |          |          |          |
 |----------|----------|----------|----------|----------|----------|----------|
 |          |          |          |          |          |          |          |
4|   6.00   |   4.00   |   9.00   |   3.00   |   1.00   |   0.00   |   6.00   |
 |          |          |          |          |          |          |          |
 |----------|----------|----------|----------|----------|----------|----------|
 |          |          |          |          |          |          |          |
5|   1.00   |   6.00   |   4.00   |   4.00   |   3.00   |   5.00   |   6.00   |
 |          |          |          |          |          |          |          |
 |----------|----------|----------|----------|----------|----------|----------|
 |          |          |          |          |          |          |          |
6|   7.00   |   8.00   |   2.00   |   1.00   |   5.00   |   0.00   |   0.00   |
 |          |          |          |          |          |          |          |
 |----------|----------|----------|----------|----------|----------|----------|


GAMES PLAYED AND RESULTS

   GAME       ROW        COLUMN     VALUE      TOTAL WINNINGS       AVERAGE
   NUMBER     PLAYED     PLAYED     OF GAME    THUS FAR             PER GAME
    1          3          4          6.000        6.000             6.000
    2          1          1          8.000       14.000             7.000
    3          3          4          6.000       20.000             6.667
    4          4          4          3.000       23.000             5.750
    5          3          1          0.000       23.000             4.600
    6          6          2          8.000       31.000             5.167
    7          1          4          3.000       34.000             4.857
    8          1          4          3.000       37.000             4.625
    9          3          4          6.000       43.000             4.778
   10          3          2          3.000       46.000             4.600
   11          3          2          3.000       49.000             4.455
   12          6          2          8.000       57.000             4.750
   13          6          4          1.000       58.000             4.462
   14          1          2          3.000       61.000             4.357
   15          4          1          6.000       67.000             4.467
   16          3          4          6.000       73.000             4.563
   17          3          4          6.000       79.000             4.647
   18          1          4          3.000       82.000             4.556
   19          6          1          7.000       89.000             4.684
   20          3          2          3.000       92.000             4.600
   21          3          2          3.000       95.000             4.524
   22          4          4          3.000       98.000             4.455
   23          3          4          6.000      104.000             4.522
   24          6          4          1.000      105.000             4.375
   25          3          4          6.000      111.000             4.440

     SUMMARY STATISTICS:
          SEED FOR RANDOM NUMBERS =      55559
          NUMBER OF GAMES PLAYED =       1000
          TOTAL WINNINGS =  4010.000

          AVERAGE PER GAME =    4.010

DESIRED ROW STRATEGY: 0.2509 0.0000 0.4467 0.1292 0.0000 0.1730
ACTUAL ROW STRATEGY:  0.2460 0.0000 0.4390 0.1460 0.0000 0.1690

DESIRED COLUMN STRATEGY: 0.2072 0.2471 0.0209 0.5247 0.0000 0.0000 0.0000
ACTUAL COLUMN STRATEGY:  0.1970 0.2600 0.0230 0.5200 0.0000 0.0000 0.0000
```

COMPUTER SUPPLEMENTED FINITE ALGEBRA

James F. Peters
St. John's University
Collegeville, Minnesota

1. Introduction

This paper offers programs of use in those portions of a course in algebraic structures devoted to introductory finite algebra. These programs are taken from a monograph by this author which will appear in the Proceedings for the 1970 Summer Institute in Computer Supplemented Instruction at the University of North Dakota. These programs are currently being used in a course in computer science and a course in elementary algebraic structures by Professor Paul P. Rowe at North Dakota State University.

The development of these programs parallels the traditional coverage of algebra topics found in Chapters 1 and 2 of Herstein (4) and that found in Chapters 1,2, and 3 of Sze-Tsen Hu (5).

To conserve space, only the flowcharts and sample outputs are presented in the body of this paper; the original programs to appear in the U.N.D. Proceedings were written in F level FORTRAN IV for the IBM 360/30. I wish to acknowledge my gratitude to Professor Warren Stenberg of the University of Minnesota, Sister Helen Marie Finn of the College of St. Mary of Omaha, Professor Gene Kemper, Mr. Conrad Dietz and Mr. Al Lindem of the University of North Dakota for their help in preparing this paper.

The mathematics required for an understanding of the programs presented here is minimal.

Suppose T = {a,b,c} an arbitrary set of three elements, and the following is a Cayley table A (sometimes called a product table) which defines an operation S on the elements of T:

	a	b	c
a	a	b	c
b	b	b	c
c	c	c	b

Now let a=1, b=2, C=3, or T ={1,2,3} and the Cayley table A becomes

	1	2	3
1	1	2	3
2	2	2	3
3	3	3	2

Notice that since the table defines an operation on the set T, the only elements appearing inside the table are those of T.

Next, using a device suggested by Professor Stenberg, we represent each of the "products" of the table using matrix notation. Thus, A(1,1) = 1, A(1,2) = 2, and so on. Observe, for example, A(2,3) actually represents the "product" of 2 "times" 3. By using this device, we can also take products inside the table and represent their products. For instance, let X = A(2,3), Y = A(3,3), then A (X,Y) represents the product of X and Y. That is,

$$A(X,Y) = A(3,2) = 3$$

In effect, we are using a fact once pointed out by Professor Joseph Rotman (3). That is, a finite group is completely known if the products of its operation table are completely known. What we have defined in the above example is not a group but something very much like it. The set T together with the operation defined on T by the Cayley table, make up what is sometimes called a groupoid. We have complete knowledge of this groupoid because we have virtually complete knowledge of its operation table.

With Professor Stenberg's device, we are able to study the products of the groupoid table quite easily and in a complete fashion. We can find out, for example, if the operation is associative. We might try doing this by hand by checking 54 different products (in the 3X3 case) or 128 different products (in the 4X4 case), and so on. By using the SEMIGP program presented here, we can check for associativity in microseconds. Hence, use of Professor Stenberg's method makes access to problems of a typical introductory course in algebra much less painful and an element of fun pops up now and then, too.

2. Program: SEMIGP

Preliminary Remarks.

To understand what this program sets out to do, the reader should keep three notions in mind. Two of these notions have already been coupled together in the Introduction to these Notes; namely the notions of set and operation. Now we should formalize these two notions and, while we are at it, introduce a third notion also previously mentioned, associativity.

Let T be a non-empty set of n elements, say $T = \{t_1, t_2, . . . t_n\}$. and let F : TxT-->T be a mapping which sends pairs of elements in T into yet other elements of T. Let x and y be elements of T, then we can write

F : (x,y) --> t in T

This is also written as

F(x,y) = t

In the rest of these Notes, however, we introduce a natural extension of Professor Stenberg's idea and write

A(x,y) = t

where A stands for the array (or matrix) used to define an operation on the set T. In effect, what we are saying is that the array A sends elements x and y to the element t in T.

We add to this that the operation A is associative if, for each x, y, and z in the set T,

A(x,(y,z)) = A((x,y),z)

when such is the case, the set T and the operation A defined on T, the pair (T,A) form what is called a semigroup. The notion of semigroup provides the key to the remainder of these Notes. It is also a cardinal notion of algebra.

Purpose.

To decide whether a given array represents an associative operation and to provide a sorting device when the output of the table-generator programs is studied.

Method.

Let the computer replace K, X, and Y by values of the set {1,2,3} and check to see if the following equality

A(X,(K,Y)) = A((X,K),Y)

holds true. This procedure can be simplified in a way suggested by Professor Stenberg. That is, we let

U = A(X,K) Z = A(K,Y)

and we then check the truth of

A(X,Z) = A(U,Y)

The flowchart for SEMIGP is shown in Figure 1. The sample output following Figure 1 exhibits an operation which fails to be associative.

Program: SORT 4X4.

This program asks the computer to first construct a 4X4 operation table and then check to see if the operation represented by the table is associative. It should be mentioned that the computer will make 4, 394, 967, 296, or 4^{16} such checks. It took the IBM 360/30 one hour and 3 minutes to find 163 beautiful 4X4 semigroups, some of which are shown in the sample output provided here. As shown in the flowchart for SORT 4X4 in Figure 2. construction of the successive 4X4 tables is made possible with nested DO loops. The check for associativity of each table is made possible by using the SEMIGP program as a subroutine.

3. Program: NML3 GP.

Preliminary Remarks.

In this program we extend the notion of right coset to richer ground, that of groups. Suppose the pair (G,A) is a semigroup where G is a non-empty set and A is an associative operation defined on G. Suppose (G,A) has an identity element e. In addition, suppose that for each x in G, there exists an element y in G such that

$$A(x,y) = A(y,x) = e.$$

That is, every element of G has an inverse. Then the semigroup (G,A) is called a group. Any subset of (G,A) which forms a group is called a subgroup. Now suppose H is a subgroup of a group G. Let Hx be the right coset of H in G for an arbitrary element x in G; let xH be the left coset of H in G. H is said to be a normal subgroup of G if Hx = xH for every x in G.

Purpose.

To search through the subgroups of a given group to determine if any of these subgroups are normal. In Figure 3 the flowchart of NML3 GP is presented. The program which the flowchart represents is intended to be illustrative, not general, since it is designed to search through all subgroups of order 3 to see if they are normal subgroups. If a subgroup H in G is a normal subgroup, the output will also exhibit the quotient group of H in G. If a subgroup H in G is not normal, the output will indicate precisely where the subgroup fails to be normal; this case is illustrated in the sample output following Figure 3.

4. Program: DISTRIB.

Preliminary Remarks.

In this program, we work with two arrays representing operations on the same set. Let X, Y, Z be elements of a set T. Let A and M be operations defined on T. Then

$$M(X,A(Y,Z)) = A(M(X,Y),M(X,Z))$$

represents the familiar notion of left distribution of M over A for X, Y, Z. Similarly, we can define right distribution of M over A, then M is said to be a distributive operation over A, if it is both left- and right-distributive for every X, Y, Z in T.

The tables available from SORT 3X3 and SORT 4X4 provide data for this program. These tables were put on tape and read from the tape by DISTRIB.

Purpose.

To sort through pairs of 3X3 tables and pairs of 4X4 tables representing semigroups to find distributive operations.

Method.

This is implicit in the preceding remarks. Suppose K, Z, and X are elements of a set T. Let A and M be operations defined on the elements of T by the given arrays. Now let

$$U = M(X,K) \quad V = M(X,Z) \quad S = M(K,X) \quad T = M(Z,X)$$

Then we determine if

$$\begin{aligned} M(X,W) &= A(U,V) &&\text{(left distributivity)} \\ M(W,X) &= A(S,T) &&\text{(right distributivity)} \end{aligned}$$

Figure 4 provides the flowchart for DISTRIB. The output following Figure 4 is of interest, since the computer indicates in what instance one operation fails to be either left or right distributive. It should be mentioned that sample outputs for both the 3X3 and 4X4 cases are presented. These outputs are especially beautiful, since they provide new examples of semirings; in the classroom, the more examples we have of this nature the better.

4. Concluding Remarks.

Many of the ideas for programs in these Notes came from an article by Professor Ward Douglas Maurer (2) entitled "Computer Experiments in Finite Algebra" which appeared in the *ACM Communications* in 1966. Professor Maurer tells of some 70 programs that were written in *MAD and FAP* for the IBM 7094 at M.I.T. Needless to say this article abounds with ideas. A partial list of the programs suggested by Maurer is provided in the Appendix of these Notes.

In January of 1969, Professor John J. Cannon (1) wrote a comprehensive survey of how computers have been used in the past three decades in this country and overseas. Certainly what Professor Cannon says should be tantalizing to anyone interested in using computers in algebra. This article could be considered provocative solely in terms of its comparison with Profess4r Maurer's article. Why? Professor Maurer comments that "the subject is still not quite respectable because it has yet to produce any concrete results -- not a single important mathematical theorem has yet been proved by computer at the time of this writing (January 1966)." On the other hand, Professor Cannon points out that L. M. Norton wrote a Ph.D. thesis (in 1966!) entitled "A heuristic program for proving theorems of group theory".

Professor Cannon mentions that the use of Norton's program (since 1966) "has found proofs for nearly 100 elementary theorems in group theory. These include the theorems concerning the intersection of subgroups, the theorem which states that if every element of a group is of order 2, then the group is Abelian, and the theorem that the kernel of a homomorphism is the identity if and only if the homomorphism is one-to-one."

Professor Cannon visualizes four types of applications of computers in group theory, each of which finds some representation in these notes:

> (1) Mechanization of tedious algebra; (2) Application of a piece of theory to particular cases; (3) Generation of examples; (4) Searching for counter-examples.

REFERENCES

1. Cannon, John J., "Computers in Group Theory: A Survey", *Communications of the ACM*, Vol. 12, No. 1, January 1969, pp. 3-12.
2. Maurer, Ward Douglas, "Computer Experiments in Finite Algebra", *Communications of the ACM*, Vol. 9, No. 8, August 1966, 598-643.
3. Rotman, Joseph, *Group Theory: An Introduction*, Allyn and Bacon, Boston 1967.
4. Herstein, I. N., *Topics in Algebra*, Blaisdell, Mass., 1964.
5. Hu, Szen-Tsen, *Elements of Modern Algebra*, Holden-Day, San Francisco, 1965.

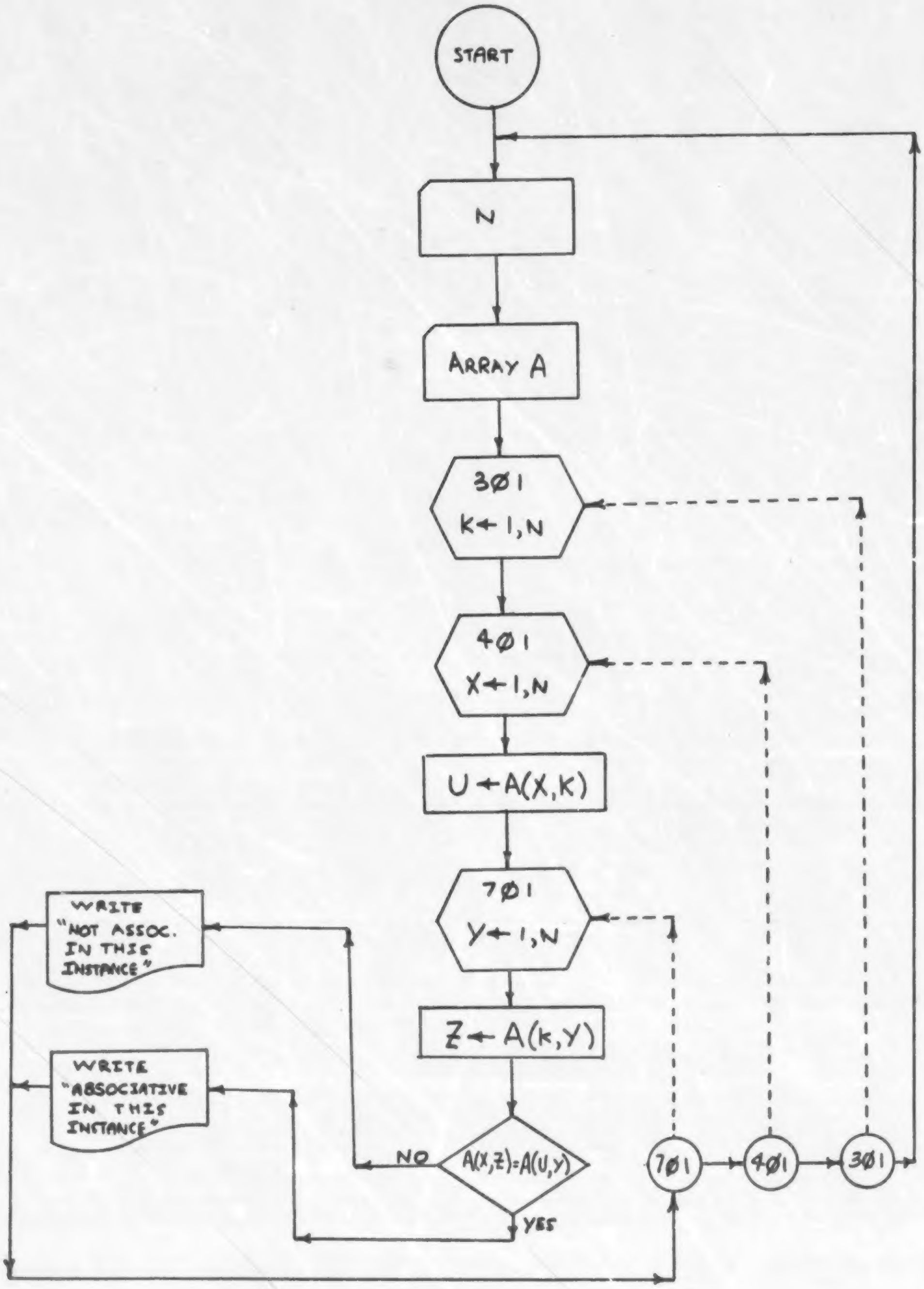

START
N
ARRAY A
3Ø1
K ← 1,N
4Ø1
X ← 1,N
U ← A(X,K)
7Ø1
Y ← 1,N
Z ← A(K,Y)
A(X,Z)=A(U,Y)
NO
YES
WRITE "NOT ASSOC. IN THIS INSTANCE"
WRITE "ASSOCIATIVE IN THIS INSTANCE"
7Ø1
4Ø1
3Ø1

Figure 1. Flowchart for SEMIGP

SAMPLE OUTPUT FOR SEMIGP.

```
   1     2     3
   2     2     1
   3     1     2
                    K       X       Y
ASSOCIATIVE         1       1       1
ASSOCIATIVE         1       1       2
ASSOCIATIVE         1       1       3
ASSOCIATIVE         2       1       1
ASSOCIATIVE         2       1       2
ASSOCIATIVE         2       1       3
ASSOCIATIVE         3       1       1
ASSOCIATIVE         3       1       2
ASSOCIATIVE         3       1       3
ASSOCIATIVE         1       2       1
ASSOCIATIVE         1       2       2
ASSOCIATIVE         1       2       3
ASSOCIATIVE         2       2       1
ASSOCIATIVE         2       2       2
NOT ASSOCIATIVE IN THIS INSTANCE      2       2       3
ASSOCIATIVE         3       2       1
NOT ASSOCIATIVE IN THIS INSTANCE      3       2       2
ASSOCIATIVE         3       2       3
ASSOCIATIVE         1       3       1
ASSOCIATIVE         1       3       2
ASSOCIATIVE         1       3       3
ASSOCIATIVE         2       3       1
ASSOCIATIVE         2       3       2
NOT ASSOCIATIVE IN THIS INSTANCE      2       3       3
ASSOCIATIVE         3       3       1
NOT ASSOCIATIVE IN THIS INSTANCE      3       3       2
```

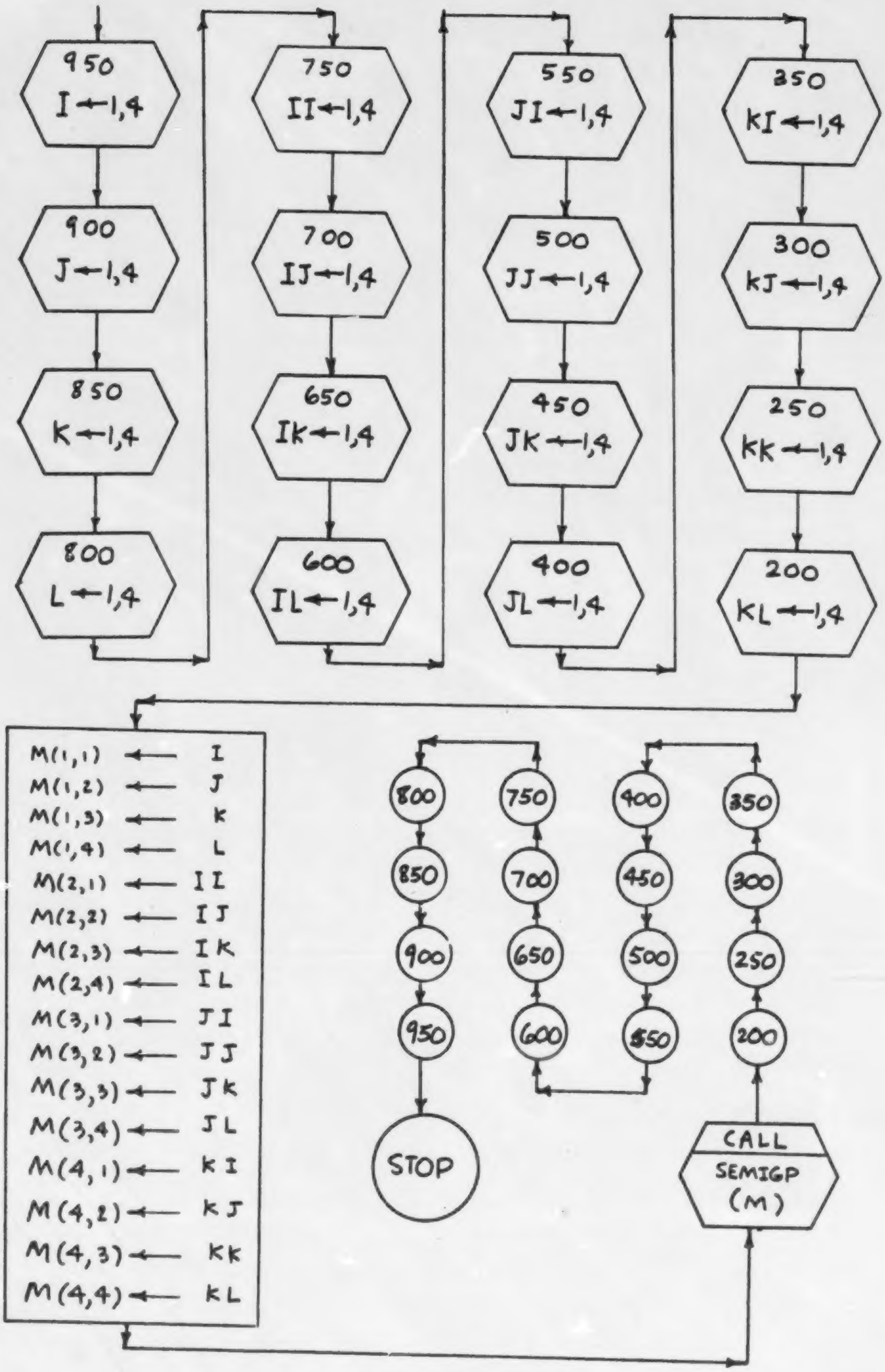

950
I ← 1,4
900
J ← 1,4
850
K ← 1,4
800
L ← 1,4
750
II ← 1,4
700
IJ ← 1,4
650
IK ← 1,4
600
IL ← 1,4
550
JI ← 1,4
500
JJ ← 1,4
450
JK ← 1,4
400
JL ← 1,4
350
KI ← 1,4
300
KJ ← 1,4
250
KK ← 1,4
200
KL ← 1,4
M(1,1) ← I
M(1,2) ← J
M(1,3) ← K
M(1,4) ← L
M(2,1) ← II
M(2,2) ← IJ
M(2,3) ← IK
M(2,4) ← IL
M(3,1) ← JI
M(3,2) ← JJ
M(3,3) ← JK
M(3,4) ← JL
M(4,1) ← KI
M(4,2) ← KJ
M(4,3) ← KK
M(4,4) ← KL
800
850
900
950
STOP
750
700
650
600
400
450
500
550
350
300
250
200
CALL
SEMIGP
(M)

Figure 2. Flowchart for SORT 4X4

SAMPLE OUTPUT FOR SORT 4X4.

```
                                        1   1   1   1
                                        1   1   1   2
                                        1   1   1   3
                                        1   1   1   4
ABOVE TABLE IS A BEAUTIFUL SEMIGP
                                        1   1   1   1
                                        1   1   1   2
                                        1   1   1   3
                                        1   1   3   4
ABOVE TABLE IS A BEAUTIFUL SEMIGP
                                        1   1   1   1
                                        1   1   1   2
                                        1   1   1   3
                                        1   2   1   4
ABOVE TABLE IS A BEAUTIFUL SEMIGP
                                        1   1   1   1
                                        1   1   1   2
                                        1   1   1   3
                                        1   2   2   4
ABOVE TABLE IS A BEAUTIFUL SEMIGP
                                        1   1   1   1
                                        1   1   1   2
                                        1   1   1   3
                                        1   2   3   4
ABOVE TABLE IS A BEAUTIFUL SEMIGP
                                        1   1   1   1
                                        1   1   1   2
                                        1   1   1   3
                                        1   3   3   4
ABOVE TABLE IS A BEAUTIFUL SEMIGP
                                        1   1   1   1
                                        1   1   1   2
                                        1   1   2   3
                                        1   2   3   4
ABOVE TABLE IS A BEAUTIFUL SEMIGP
                                        1   1   1   1
                                        1   1   1   2
                                        1   1   3   1
                                        1   1   1   4
ABOVE TABLE IS A BEAUTIFUL SEMIGP
                                        1   1   1   1
                                        1   1   1   2
                                        1   1   3   1
                                        1   2   1   4
ABOVE TABLE IS A BEAUTIFUL SEMIGP
                                        1   1   1   1
                                        1   1   1   2
                                        1   1   3   3
                                        1   1   3   4
ABOVE TABLE IS A BEAUTIFUL SEMIGP
```

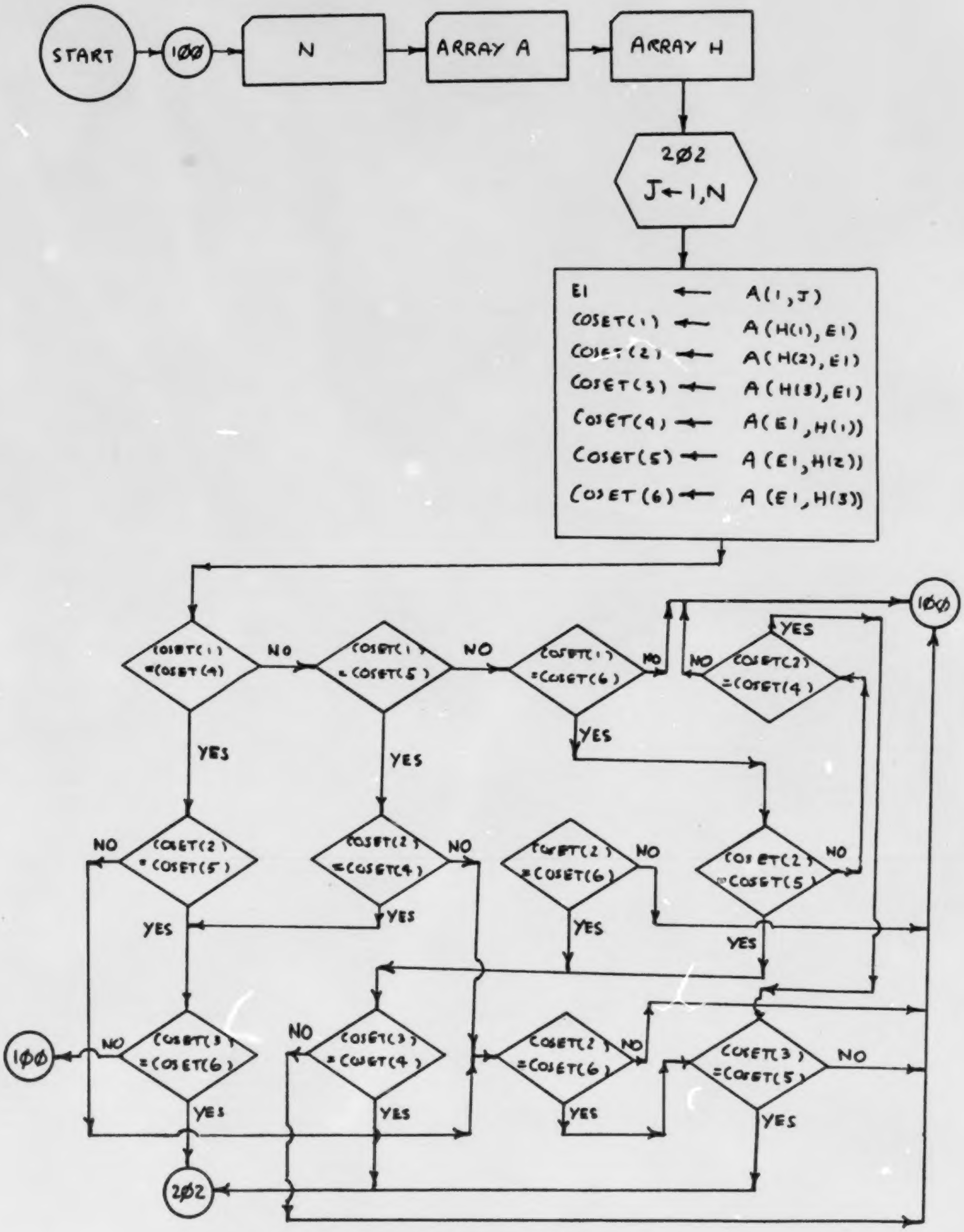

START
100
N
ARRAY A
ARRAY H
202
J ← 1,N
E1 ← A(1,J)
COSET(1) ← A(H(1),E1)
COSET(2) ← A(H(2),E1)
COSET(3) ← A(H(3),E1)
COSET(4) ← A(E1,H(1))
COSET(5) ← A(E1,H(2))
COSET(6) ← A(E1,H(3))
COSET(1) =COSET(4)
NO
COSET(1) =COSET(5)
NO
COSET(1) =COSET(6)
NO
100
YES
NO
COSET(2) =COSET(4)
YES
YES
YES
NO
COSET(2) =COSET(5)
COSET(2) =COSET(4)
NO
COSET(2) =COSET(6)
NO
COSET(2) =COSET(5)
NO
YES
YES
YES
YES
100
NO
COSET(3) =COSET(6)
NO
COSET(3) =COSET(4)
COSET(2) =COSET(6)
NO
COSET(3) =COSET(5)
NO
YES
YES
YES
YES
202

Figure 3. Flowchart for NML3 GP

SAMPLE OUTPUT FOR NML3 GP

```
    GROUP TABLE
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
 2  7  4  5  6  3  8  1 10 11 12  9 14 15 16 13 18 19 20 17 22 23 24 21
 3  9 13 17 10  2 21 14  6 20 22  7  1 24 18  4 16 23 11  5 12 19 15  8
 4 10 14 18 11  7 22 15  3 17 23  8  2 21 19  5 13 24 12  6  9 20 16  1
 5 11 15 19 12  8 23 16  4 18 24  1  7 22 20  6 14 21  9  3 10 17 13  2
 6 12 16 20  9  1 24 13  5 19 21  2  8 23 17  3 15 22 10  4 11 18 14  7
 7  8  5  6  3  4  1  2 11 12  9 10 15 16 13 14 19 20 17 18 23 24 21 22
 8  1  6  3  4  5  2  7 12  9 10 11 16 13 14 15 20 17 18 19 24 21 22 23
 9 21 17 10  2 13 14  3 20 22  7  6 24 18  4  1 23 11  5 16 19 15  8 12
10 22 18 11  7 14 15  4 17 23  8  3 21 19  5  2 24 12  6 13 20 16  1  9
11 23 19 12  8 15 16  5 18 24  1  4 22 20  6  7 21  9  3 14 17 13  2 10
12 24 20  9  1 16 13  6 19 21  2  5 23 17  3  8 22 10  4 15 18 14  7 11
13  6  1 16 20  9 12 24  2  5 19 21  3  8 23 17  4 15 22 10  7 11 18 14
14  3  2 13 17 10  9 21  7  6 20 22  4  1 24 18  5 16 23 11  8 12 19 15
15  4  7 14 18 11 10 22  8  3 17 23  5  2 21 19  6 13 24 12  1  9 20 16
16  5  8 15 19 12 11 23  1  4 18 24  6  7 22 20  3 14 21  9  2 10 17 13
17 20 24 23 22 21 19 18 13 16 15 14  9 12 11 10  1  8  7  2  6  5  4  3
18 17 20 24 23 22 20 19 14 13 16 15 10  9 12 11  2  1  8  7  3  6  5  4
19 17 22 21 24 23 17 20 15 14 13 16 11 10  9 12  7  2  1  8  4  3  6  5
20 18 23 22 21 24 18 17 16 15 14 13 12 11 10  9  8  7  2  1  5  4  3  6
21 14 10  2 13 17  3  9 22  7  6 20 18  4  1 24 11  5 16 23 15  8 12 19
22 15 11  7 14 18  4 10 23  8  3 17 19  5  2 21 12  6 13 24 16  1  9 20
23 16 12  8 15 19  5 11 24  1  4 18 20  6  7 22  9  3 14 21 13  2 10 17
24 13  9  1 16 20  6 12 21  2  5 19 17  3  8 23 10  4 15 22 14  7 11 18

SUBGROUP H
  1      3       13

H(1)A(I,J)     H(2)A(I,J)                   H(3)A(I,J)   A(I,J)H(1)   A(I,J)H(2)   A(I,J)H(3)

     1                 3                         13          1           3            13

H(1)A(I,J)     H(2)A(I,J)                   H(3)A(I,J)   A(I,J)H(1)   A(I,J)H(2)   A(I,J)H(3)

     2                 9                          6          2           4            14

     H IS NOT NORMAL IN G
```

271

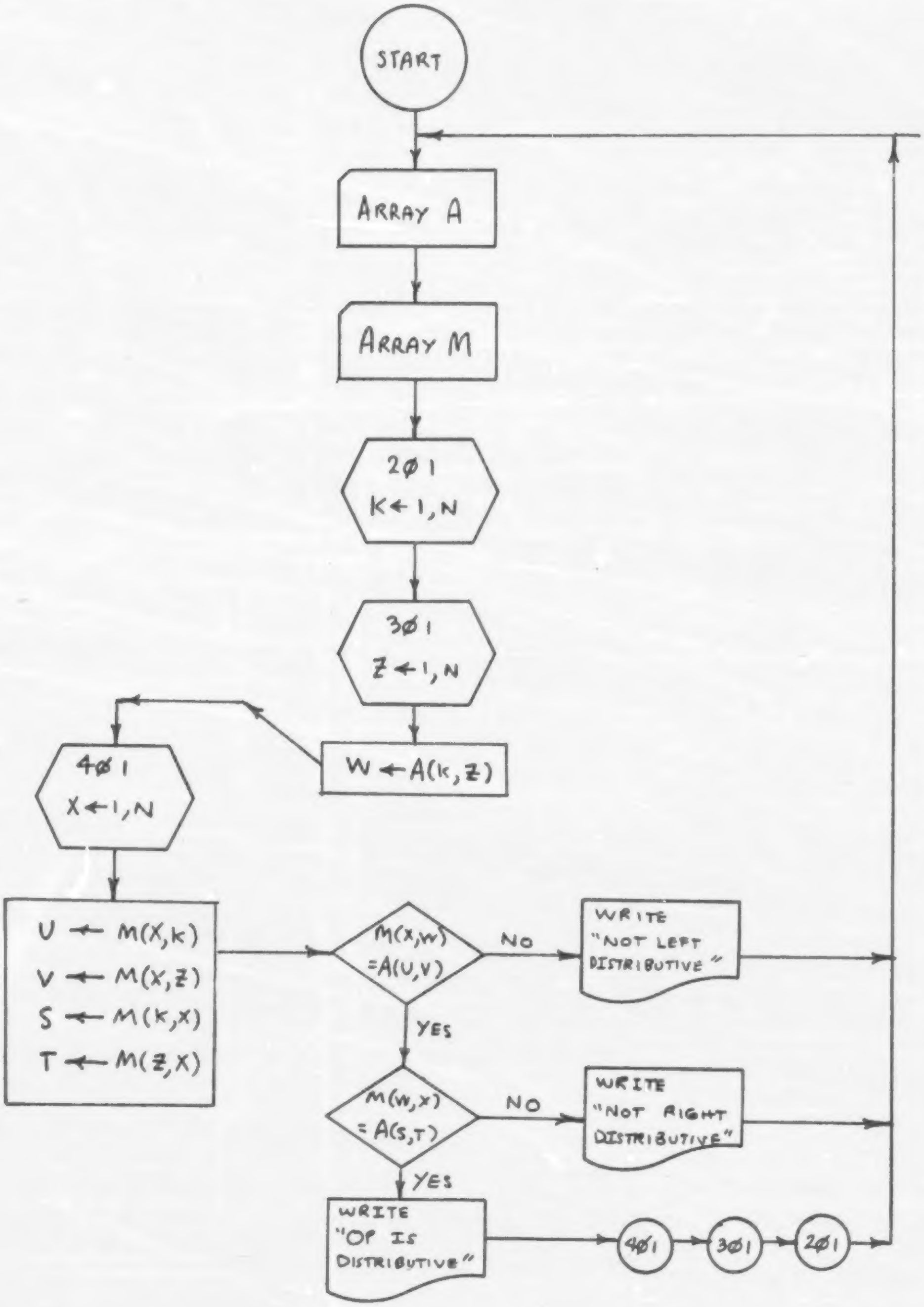

START
ARRAY A
ARRAY M
2Ø1
K←1,N
3Ø1
Z←1,N
W←A(K,Z)
4Ø1
X←1,N
U ← M(X,K)
V ← M(X,Z)
S ← M(K,X)
T ← M(Z,X)
M(X,W) =A(U,V)
NO
WRITE "NOT LEFT DISTRIBUTIVE"
YES
M(W,X) = A(S,T)
NO
WRITE "NOT RIGHT DISTRIBUTIVE"
YES
WRITE "OP IS DISTRIBUTIVE"
4Ø1
3Ø1
2Ø1

Figure 4. Flowchart for DISTRIB.

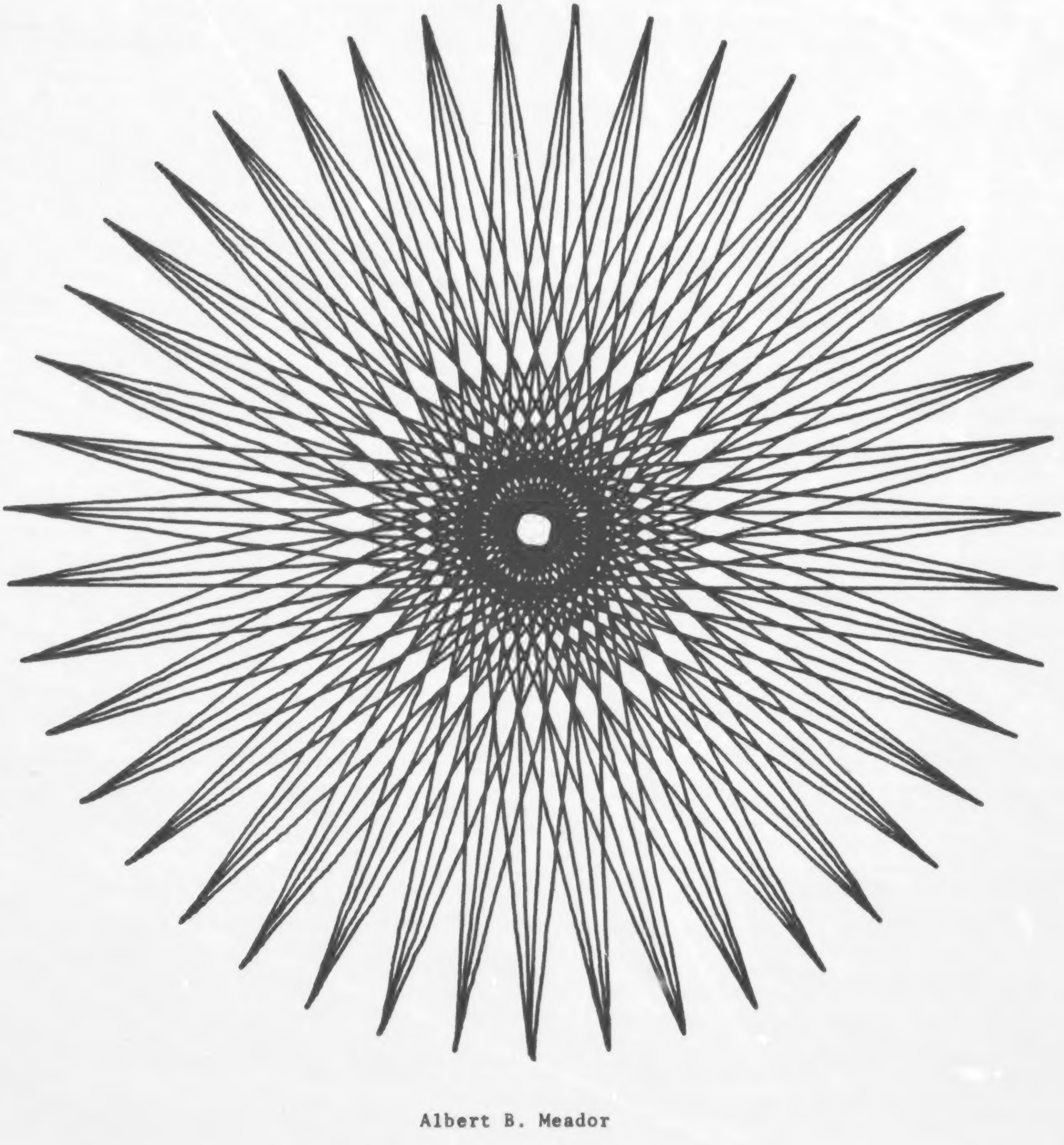
SAMPLE OUTPUT FOR DISTRIB.

```
                                               1     2     1       1     2     1
                                               1     2     1       1     2     2
                                               1     2     3       1     2     3

NOT LEFT-DISTRIBUTIVE           2           1           3
                                               1     2     1       1     2     1
                                               2     1     2       2     1     2
                                               1     2     1       1     2     3

NOT LEFT-DISTRIBUTIVE           2           1           1
                                               1     2     1       1     2     1
                                               2     2     2       2     2     2
                                               1     2     1       1     2     3

OP IS DISTRIBUTIVE

                                               1     2     1       1     2     2
                                               2     2     2       1     2     2
                                               3     2     3       1     2     2

NOT LEFT-DISTRIBUTIVE           1           1           3
                                               1     2     2       1     2     2
                                               1     2     2       2     1     1
                                               1     2     3       2     1     1

OP IS DISTRIBUTIVE

                                               1     2     2       1     2     2
                                               2     2     2       2     2     2
                                               2     2     2       2     2     2

OP IS DISTRIBUTIVE

                                               1     2     2       1     2     2
                                               2     2     2       2     2     2
                                               3     2     2       3     3     3

NOT LEFT-DISTRIBUTIVE           3           1           1
                                               1     2     3       1     2     3
                                               1     2     3       1     2     3
                                               1     2     3       3     3     3

OP IS DISTRIBUTVE

                                               1     2     3       1     2     3
                                               2     1     3       2     2     2
                                               3     3     3       1     2     3
```

SAMPLE OUTPUT FOR DISTRIB.

```
                                       1  1    1   1     1   1   1   1
                                       1  1    1   2     1   1   1   2
                                       1  2    3   1     1   2   3   2
                                       1  1    1   4     1   1   1   4

NOT LEFT-DISTRIBUTIVE        3      2       4
                                       1  1    1   1     1   1   1   1
                                       1  1    1   2     1   1   1   2
                                       1  2    3   3     3   3   3   3
                                       1  2    3   4     1   1   1   4

NOT RIGHT-DISTRIBUTIVE       1      3       4
                                       1  1    1   1     1   1   1   1
                                       1  1    1   2     1   1   1   2
                                       3  3    3   3     3   3   3   3
                                       1  1    3   4     1   2   1   4

NOT RIGHT DISTRIBUTIVE       1      4       3
                                       1  1    1   1     1   1   1   1
                                       1  1    1   2     1   1   1   2
                                       3  3    3   3     3   3   3   3
                                       1  2    3   4     3   3   3   4

NOT RIGHT DISTRIBUTIVE       1      4       1
                                       1  1    1   1     1   1   1   1
                                       1  1    1   3     1   1   1   3
                                       1  1    1   1     1   1   1   1
                                       1  1    1   1     1   1   1   3

OP IS DISTRIBUTIVE

                                       1  1    1   1     1   1   1   1
                                       1  1    1   3     1   1   1   3
                                       1  1    1   1     1   1   1   1
                                       1  3    1   1     1   3   1   2

OP IS DISTRIBUTIVE

                                       1  1    1   1     1   1   1   1
                                       1  1    1   3     1   1   1   3
                                       1  1    1   1     1   1   1   3
                                       1  3    1   3     1   1   1   4

NOT RIGHT-DISTRIBUTIVE       4      2       4
                                       1  1    1   1     1   1   1   1
                                       1  1    1   3     1   1   1   3
                                       1  1    1   3     1   1   1   3
                                       1  2    3   4     1   3   3   4
```

273



Albert B. Meador

DESIGN OF A COURSE IN SOCIAL APPLICATIONS OF COMPUTING

Andrew R. Molnar*
National Science Foundation
Washington, D.C. 20550

Our society is said to be in a transition from one based upon industrial production to one based upon knowledge. It is estimated that by the end of the decade, computer and computer related activities will account for as much as one-third to one-half of our gross-national-product. Everyone in our society will be affected, in one way or another, by the computer. The computer offers us the capacity to significantly improve the quality of life and to solve many of the critical problems facing our society (1).

In addition to its potential impact upon society, the computer can markedly alter the role of education in the knowledge society. In this period of transition it is important to commence the dialogue on what the role of university based computing should be in ameliorating the problems of society.

It has been said that the three main functions of modern colleges and universities are instruction, research, and social service. While instruction and research have historically received the most attention, of late, social service has emerged as a significant force. Many schools are attempting to redesign their curricula in order to make them more relevant to modern social needs. Many institutions are rededicating their resources to the study and solution of social problems. While the computer has played an important role in research, and more recently in instruction, the use of university based computers in the support of social service applications to societal problems has been virtually nonexistent. However, it is becoming clear that the university is destined to play an active role in the fashioning of solutions to complex social and technological problems and will be a basic industry in the knowledge society.

1. Instruction. The computer can alter how and what we teach. With the advent of the "information explosion", some have advocated the reorganization of education around the concepts of "broad-gathering" and "narrow-casting". That is, it is proposed that large educational resource information centers be established to search out all forms of information and instructional materials and store them in large computer utilities which can make them rapidly available on demand to remote educational sites or even to individual homes through modern telecommunications. With the development of low-cost, home television which is capable of using film or video-tape cassettes, the use of computer terminals can make it possible for an individual to have a complete individualized education delivered directly to his living room. Further developments could make low-cost television and computer terminals as common in the home and office as central heating or electricity and could be offered at rates comparable to those of present day utilities.

A number of colleges and universities are abandoning the classroom and specialized disciplines and have initiated experimental programs organized around problems in the community and society. It will be possible in the near future to obtain a degree by solving urban problems through computer applications for a specific region or locale. In the past, complex social problems were simplified so they could be discussed in general terms in the classroom. Many argue that the simplification reduced the relevance of instruction and increased the difficulty of transferring what was learned in the classroom to "real problems". The forte of the computer is complexity and the rapid processing of large amounts of data. By challenging the student with real problems and by placing him in direct contact with real data, it is possible to increase both the relevancy and the meaning of education.

2. Research. More and more, the complex problems of society and our ability to cope with them through Government are becoming the concern of the university. For unlike the problems of the market place that reach equilibrium through such mechanisms as supply and demand, we find that social and technological problems do not necessarily resolve themselves without conscious thought and intervention. Likewise, Government efficiency is not regulated by price mechanisms or profits, and in the day-to-day operations of Government, it is not possible to consider all alternatives, to evaluate all actions and assess all effects. Therefore, there is a need for the careful, systematic analysis of complex social and environmental problems with the aid of analytic tools such as the computer. This type of information can do much to assist policy makers to more effectively carry out their missions.

The number of computers in Government has grown from 403 in 1959 to 5,277 in 1970 (2). It is not inconceivable that in the near future through the Public Information Act, the Federal Government and possibly state and local governments could make immediately available computer tapes for analyses. Such complex questions as the National Budget, the redistricting of state representatives boundaries or zoning modifications before local governments could be analyzed and their impact considered prior to decision. National, regional, and local data bases could be established at minimal cost. Universities with a commitment to social service could provide the faculty and students, as part of their academic training, to analyze and report their research as a public service.

3. Social Service. If the existing raw computer-capacity of the nation's universities was devoted to applied problems, significant changes could be fashioned. For example, analysis of local traffic and transportation problems could produce significant changes in the allocation of resources. Computer centers could monitor and analyze epidemic trends derived from medical records within a region by having selected doctors phone in their case record statistics on a daily basis. Samples from environmental sensing devices could feed into a regional computer and provide data for research as well as act as an environmental alert system. On-line computer based attitude analyses on selected topics could provide policy makers with accurate and timely information on complex issues.

Frequently, high level policy makers inquire as to the potential of computers in solving complex social problems. The quantity of well-thought-out ideas is relatively small and seldom documented. What is needed is to explore the feasibility of applying computer techniques to assist in the solution of complex problems and to identify the university's part in this process. What will the role of the university be in the knowledge society? How can computer-based education systems be used to aid in the solution of complex social and technological problems? Now is the time to ignore the constraints of resources and to commence the discussion on what could be. Now is the time to consider the future.

FOOTNOTES

1. Andrew R. Molnar, "Computer-Based Instruction and the Knowledge Society", Speech delivered at the Conference on "Computer in Education" sponsored by the Centro Studi e Applicazioni in Tecnologie Avanzate, Universita' Degli Studi di Bari, Bari, Italy, October 1-3, 1970.

2. General Services Administration. "Fiscal Year 1970 Inventory of Automatic Data Processing Equipment in the United States Government", (Washington, U. S. Government Printing Office, 1970).

* The views expressed are those of the author and do not necessarily represent those of the National Science Foundation. This paper is based upon comments delivered at the "Conference on Computers in the Undergraduate Curricula" at Dartmouth College, Hanover, N.H., June 23-25, 1971.

PLANNING AN EXPERIMENTAL SEMINAR ON "COMPUTERS, SOCIETY, AND THE FUTURE"

Craig Decker
University of Virginia
Charlottesville, Virginia

Introduction

An interdisciplinary seminar with the ambitious title "Computers, Society, and the Future" will be offered at the University of Virginia this spring. This paper will outline the organization and objectives of the course.

As with most computer related activities, the seminar may be discussed on two levels--the "semantic" or course content level and the "syntactical" or course organization and planning level.

The Semantic Level

The general objectives on this level were to 1) survey a wide spectrum of computer applications; e.g., in law, medicine, business, education, government and urban design, so that participants would have a basis on which to 2) assess the social and philosophical impact of the computer revolution, and 3) develop computer models of social, political and ecological systems.

In order to find out where to place emphasis, a questionnaire was mailed to anyone expressing interest in the course (see Appendix I). Table I shows the average ratings of possible topics (on a 1 to 10 scale, 10 being highest interest), and the number of people rating each above 5½ out of a possible 21. It is interesting that the topics which actually dealt with developing techniques such as information retrieval and optimization techniques, consistently received higher ratings than the survey of philosophical topics.

Some fundamental questions which we hope to deal with are:[2]

1. What are realistic models of society which highlight non-optimum subsystems?
2. Will computers and modern technology ultimately be used to control or liberate men?
3. How will the computer revolution change man's view of himself?
4. How can computers be used to simulate social systems so that the future impact of present policy decisions may be predicted?

The first three questions are subjective and philosophical. They can and have been debated at great length without resolution. What is important when considering these questions is not what particular scientists or writers answer, but how several answers interact and interrelate. For example, Table II presents seven answers to the first question. Through reading a combination of these authors, one gets a much better grasp of social problems and processes than if he read the equivalent amount by just one of them. When the reader finds dissimilar points, he must consider each point more deeply, when he finds similar points he may abstract a general principle to be tested on other writings.

People express much concern and anxiety over the second question. Discussing the question is most frustrating, because it invariantly leads to a cyclic process from which no one leaves feeling he's come to grips with the problem. Computers provide a great potential for both control and liberation as indicated by Table III. Even each item in Table III has both liberating and controlling potentials.

The fourth question can be dealt with more objectively. Much research has been done in the area of political and social simulations, including Pool (1), Forrester (2,3), Guetzkow (4), Loehlin (5), and Cherryholmes (6). We hope to present a cross-section of this research and develop some actual simulation techniques, such as game theory, linear programming, Monte Carlo methods, and heuristic programming.

One might summarize the seminar on the semantic level as finding ways of "helping computers help people".

The Syntactical Level

Traditionally, the teacher has been solely responsible for course organization and the processes through which semantic information is transmitted. It is a basic premise of this seminar that participants should be involved not only on the semantic level but also on the syntactical level. This is due in part to the fact that the seminar was organized and is being run chiefly by a fourth year engineering student[3], hence there really is no professor. General objectives on the syntactical level are:

1. To make education enjoyable and relevant, i.e. applicable to current issues and problems,
2. To create a cooperative atmosphere in which participants learn from each other,
3. To instill a sense of involvement so that participants take the responsibility for learning without tests[4] or grades,
4. To encourage the interaction among persons of different disciplines, e.g. engineers and liberal arts students, and
5. To integrate technical and philosophical material into the same course, i.e. to decompartmentalize education.

These are not new goals (with the exception of (5) perhaps) and in many ways are more formidable than the fundamental questions on the semantic level. There are, however, several advantages which the seminar has over traditional courses. First, since participants are involved on the syntactical level, the whole class rather than just the teacher will be working towards the goals. Secondly, feedback and direction from the participants will be easier since the "teacher" is a peer. Thirdly, the student "teacher" will devote full time to the seminar, hence he will have time to deal with participants on an individual level. Finally, the subject itself, "Computers, Society, and the Future", lends itself to interdisciplinary discussion, and both technical and non-technical material.

Table IV shows the average ratings (on a 1 through 10 scale) and times above 5 (out of 21 possible) for possible class activities listed on the questionnaire (Appendix I). Simulation games which might be used in the seminar include: SMOG, ECOLOGY, THE CITIES GAME, BLACKS AND WHITES, SIMSOC, STAPOL, and CITIES II (7).

Other possible activities include informal luncheon and dinner meetings with interesting speakers; sitting in on Senator Ervin's Subcommittee hearings on invasion of privacy, and trips to institutes and universities actively engaged in social simulation research, e.g. Brookings Institute, System Development Corporation, and Mitre Corporation (Washington) or Abt Associates, Urban Systems, Inc., and Project Cambridge (Cambridge, Massachusetts).

Table V displays the results from the part of the questionnaire on optimum course structure and the actual seminar makeup. The most significant agreement was on the question of students grading themselves. It was decided that participants should grade themselves using guidelines proposed by the coordinators. Also, extensive use of guest speakers was planned, including professors from computer science, political science, biomedical engineering, business, law, education, medicine, and religion.

It was felt that in order to understand aspects of computer simulations, artificial intelligence, computer aided instruction, and other areas we wished to cover, the participants should attempt to write some simple programs in these areas. During the first week of classes, persons unfamiliar with BASIC will be taught the language by those in the class who have previously used it. Table VI gives a list of possible programs which may be assigned to be run on the Engineering School's Time Shared Hewlett-Packard computer. Appendices II and III contain writeups on the first two programs. Since most participants will be unfamiliar with statement calculus, a computer aided instruction program has been written in BASIC to teach basic concepts and also how to do string manipulations in BASIC (an area neglected in most computer science courses).

The importance of the syntactical level and its interrelation to the semantic level can be summarized by McLuhan's now famous phrase "the medium is the message".

Group Projects

Major emphasis will be placed on group projects intended to utilize computers to improve people's lives. Seven possible projects are listed in Table VII along with the ratings they received on the questionnaires. Other suggestions were: an automated food co-op for students and poor, computerized consumer protection service for Charlottesville buyers, a world game extension group (8), a computer simulation of the student health center[5], a computerized growth model of the University including an index of the quality of education, computer algorithms for deciding land allocation, a program to match roommates according to personality questionnaires, and a computerized counselling system for students (9). Planning of group projects will begin the third or fourth week of the course. Average group size will probably be three to five. Care must be taken to see that the projects are stimulating but not unmanageable and that the members of each group have the proper mixture of expertise to accomplish the task.

During the middle of May, a symposium will be held, at which each group will present their project. Outside speakers will be invited to give keynote addresses on the social and philosophical impact of computers, and the public will be invited. If possible a monetary prize will be given for the best presentation. It is hoped that the projects might serve as a nucleus for larger research endeavors undertaken by the participants in later years.

The Course in Progress

At this writing the seminar has been in progress for three weeks. There are two sections--one with 18 members which meets one evening a week, and one with 14 members that meets two afternoons a week. The evening section is currently reading from Pylyshyn's Perspectives on the Computer Revolution, while the afternoon section is using Martin and Norman's The Computerized Society.[6] Other readings have included articles from Forum Magazine and Forrester's "Counterintuitive Behavior of Social Systems".[7]

Before classes started an optional sensitivity training workshop/spaghetti dinner was held.[8] The session seemed to be generally enjoyable to the participants (about half the total enrollment). It is too early to tell if the workshop had a more profound effect than to acquaint participants with each other (though this alone would be worth the effort). Also we have already had one informal luncheon at which Dr. Donald Wright spoke about computer applications in medicine. Members of the class will participate in a CITIES II simulation workshop for state officials at Williamsburg, Virginia, in March.

At the first evening class, the computer's potential for both liberation and control was discussed. To allow each participant more "air time", we broke into three smaller groups midway through the class and then summarized the separate discussions at the end. The discussion was stimulating, but left one rather frustrated by the complexity and subjectivity of the issue. At the second meeting a series of eight poems were read, and the class was asked which were written by computers.[9] The first poem was thought by everyone to be computer made, but was actually written by E.E. Cummings (11). The second, also by E. E. Cummings, was correctly classified by everyone. The rest of the votes were split about 50-50. After this diversion, the dichotomy between the semantic and syntactical levels of the course were presented and Dr. Coleman Clarke from Religion spoke on man as an information processing unit. After coffee and cookies we again broke into smaller groups to discuss readings from Pylyshyn (semantic) or the direction of the course (syntactical). It was decided that group projects should be discussed the following week, which they were. Two projects, the children's book on computers[10] and a computerized roommate matching service are already planned, and two more, modeling the University and applying allocation algorithms to the layout of the Engineering School, are in the planning stage.

The afternoon section has been somewhat more structured thus far. We have had a discussion of Forrester's paper, a lecture on simple self-organizing system, remarks by Law Professor William McKee on computer applications in law, and student led discussion on computer applications in law enforcement, financial accounting, the home, corporations, medicine, transportation, and the cities (the first 12 chapters of Martin). Most participants have finished the penny match program and are working on the computer-aided instruction package for the WFF identifier.

Table VIII shows participant response to statements designed to assess attitudes toward computers and modern technology (each statement was rated from 1 to 5, 5 showing strong agreement). Averages from five participants with technical backgrounds (TECH column) and five with nontechnical backgrounds (NON column) were compared with the overall averages (TOTAL column). It was found that the nontechnical group was consistently more optimistic than the technical group about what technology could do and its positive effects on society, particularly on the statement about accelerating technological advancements to insure a high quality of life.[11] The technical group disagreed more strongly with this statement than any other, while the nontechnical group agreed more strongly than with any other statement. It will be interesting to see what effects if any the seminar has on participants' attitudes.

TABLE I

POSSIBLE TOPICS

Topics	Average Rating	Times Rated Above Five
Information Retrieval Techniques	7.50	18
Application of Computers to Urban Problems	7.45	18
Game Theory and Optimization	7.40	17
Computer Simulation of Social and Political Systems	7.15	16
Computer Simulation of Intelligence and Personality	7.10	17
Computer Aided Instruction	6.65	13
Cybernetics	6.55	14
World Simulation for Allocation of Resources	6.15	12
Applications of Computers in Medicine	5.95	12
Applications in Law and Law Enforcement	5.95	12
Automation and the "Leisure Society"	5.65	10
Computer Uses in the Home	4.80	9
Computer Art, Music, Poetry	4.40	7
Computers and Religion	4.20	7

TABLE II

MODELS OF WHAT'S WRONG WITH OUR SOCIETY

Who	What	Where
Paul Ehrlich	over-population	The Population Bomb
Laurence Peter	"in a hierarchy, every employee tends to rise to his own level of incompetence."	The Peter Principle
Stafford Beer	there are no meta-systems to control society's esoteric boxes.	"The Management of Complexity"[12]
Jay Forrester	social systems contain multiple nonlinear feedback loops which the human mind is not equipped to understand, hence their behavior is counterintuitive.	Urban Dynamics World Dynamics
Buckminster Fuller	people still believe in an economy of scarcity, even though through synergy, i.e., doing more and more with less and less, we can bring everyone in the world a "bare maximum" quality of life.	Operation Manual for Spaceship Earth
Charles Reich	there is a discrepancy between the realities of our society and our beliefs about then or "consciousness". Consciousness lags reality.	The Greening of America
Daniel Boorstin	the ideals of our society are "self-liquidating", i.e., once they are reached they cease to be ideals.	"The Self-Liquidating Ideals"[12]

TABLE III

COMPUTER POTENTIALS FOR CONTROL AND LIBERATION

Computerized Data Banks	Used to pressure people into conformity.	Allows government to deal more efficiently and completely with people's social needs on an individual basis.
Automation	A very few could control the means of production and hence have power over the rest.	Frees men from boring, repetitive tasks.
Computerized Financial Accounting	A complete record of a persons activities would be present in the system for authorities to tap.	Frees men from the boring task of bill paying, book balancing, and income tax return. Reduces crime
Computer Simulation	Simulations of personality could be used to decide the best way to persuade people to do something.	Allows man to foresee the consequences of different decisions, and hence control his future.
National Referendum		A direct democracy as found in the "Golden Age of Greece".
Computer Identification Using Voice Prints or Retina Structure	Could be distributed throughout cities so that the movements of anyone could be traced.	Necessary for national referendum and financial accounting.
Vehicle Monitoring System	Allows authorities to trace vehicle movement and hence activities throughout day.	Would provide more efficient traffic flow.

TABLE IV

Activities	Average Rating	Times Rated Above Five
Playing environmental, urban planning or political simulation games	7.60	17
Using Computer Aided Instruction Package	6.75	15
Demonstration of using "interactive graphics" in architectural design	6.75	14
Going to a conference of computer professionals	5.80	13
Sensitivity Training workshop to establish a sense of cooperation and community	5.30	11
Listening to computer music and watching computer made films	4.55	9

TABLE V

COURSE STRUCTURE

Questionnaire Results:

	High	Average	Low
% of time devoted to lectures	60	30.2	15
% of time devoted to learning techniques	50	30	0
% of non-engineers in seminar	75	53.5	37
# of participants	30	14.5	5
# of guest speakers	20	4	1
# of pages of reading/week	150	76	50
# of hours devoted to group project	43	19.5	5
Should students grade themselves?	18-yes		3-no

Approximate Seminar Make-Up:

College		11
Mathematics	2	
Psychology	1	
Government	2	
Environmental Science	1	
Religion	1	
Other	4	
Education		1
Engineering		11
Commerce		2
Architecture		1
Law		1
Medicine		1
Mid-Career Executives		1
		29

1st year	4
2nd year	4
3rd year	4
4th & 5th year	11
Fac. and Grad.	6
	29

TABLE VI

POSSIBLE PROGRAMS TO BE WRITTEN IN "BASIC"

1. A program to identify WFFS (well formed formulas).
2. An adaptive penny match program which adjusts its play to the bias of its opponent.
3. A simulation of telephone answering at the student union (Newcomb Hall Desk).
4. A program which searches the decision tree for a stick game (there are initially M sticks, players alternately take one to N sticks, the player who takes the last stick loses).
5. A program to find the optimum stand on issues given the incumbents records and voters' utility function on each issue.
6. A simple computer model of a politician's personality.
7. A program which learns from experience to play Tic-Tac-Toe.
8. A short computer-aided instruction program.
9. A computer simulation of a simple resource allocation game.
10. Given the data from questionnaires (summarized in Tables 1,4,5, and 7), decide how to divide students into sections and which topics to teach in each section.
11. A computerized poetry writer.

TABLE VII

	Average[13] Rating	Times Rated Above Five	Times Most Interesting
Program to simulate a simple social, political, or ecological system.	7.93	12	11
Simulation of traffic flow at the University.	6.53	11	2
Planning information retrieval system for students to find out "where things are at".	6.53	10	5
Computerized Voter-Candidate Matching Service for uninformed voters.	5.87	8	
Assessment of how well large data banks can be protected to insure privacy.	5.47	6	1
Designing algorithms for computer art or poetry.	5.33	8	1
Writing and illustrating a children's book about "Chuck the Computer".	4.13	6	

TABLE VIII

STATEMENTS DESIGNED TO ASSESS PARTICIPANTS' ATTITUDES ABOUT COMPUTERS

Statement	Yes	No	Average Tech	Average Non	Total
Computers will be more intelligent than men within twenty years.	2	9	2.2	2.6	2.4
Computers can do only what they are told to do.	10	7	3.4	2.8	3.35
Computer professionals are generally conservative and insensitive to social issues.	3	12	2.6	2.0	2.2
If a technique is there, man will use it, hence technology controls man rather than man controlling technology.	5	14	2.6	2.4	2.3
It would be desirable to have nationwide referendums on all important issues.	11	8	3.0	3.2	3.35
Automation will make the unemployment problem increasingly acute.	11	7	3.4	3.2	3.1
Technological advancements must be accelerated in order to insure all people a high quality of life.	5	9	2.0	3.4	2.65
The danger of invasion of privacy outweighs the benefits of large data banks.	5	11	2.6	2.4	2.45
Direct augmentation of human mental capacity by interconnection of brain with computer would be desirable.	4	6	3.6	3.2	2.85

NOTES AND REFERENCES

1. Pool, Ithiel de Sola, Robert Abelson, and Samuel Popkins, "Candidates, Issues, and Strategies: A Computer Simulation of the 1960 and 1964 Presidential Elections, M.I.T. Press, Cambridge, Massachusetts, 1965.
2. Forrester, Jay W., *Principles of Systems*, Wright-Allen Press, Cambridge, Massachusetts, 1968.
3. ______, *Urban Dynamics*, M.I.T. Press, Cambridge, Massachusetts, 1969.
4. Guetzkow, Harold, *Simulation in Social Science: Readings*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1962.
5. Loehlin, John C., *Computer Models of Personality*, Random House, Inc., New York, 1968.
6. Cherryholmes, Cleo H., and Michael J. Shapiro, "Representatives and Roll Calls: A Computer Simulation of Voting in the Eighty-Eighth Congress".
7. SMOG and ECOLOGY are Urban Systems, Inc., games (Cambridge, Massachusetts). THE CITIES GAME and BLACKS AND WHITES are *Psychology Today* games (order through magazine). A manual on SIMSOC by William Gamson is available through the Free Press, New York. STAPOL was developed at the Institute for the Future (Middletown, Connecticut). CITIES II was developed by Environmetrics in Washington, D. C.
8. For information on World Game write: Tom Turner, Southern Illinois University, P. O. Box 99, Carbondale, Illinois 62901.
9. See Cogswell, J. F., C. P. Donahoe, D. P. Estavan, and B. A. Rosenquist, "The Design of a Man-Machine Counselling System", AP-21576, Systems Development Corporation, Santa Monica, California, 1966.
10. The computerized poetry was done by Marie Borroff and appeared in *Yale Alumni Magazine*, January, 1971. The first poem by E. E. Cummings began:

    "Notice the convulsed orange inch of moon
    Perching on this silver minute of evening..."

ACKNOWLEDGEMENTS

I wish to thank Dr. James G. Simmonds, Associate Professor of Applied Mathematics, University of Virginia. It takes a special kind of person to devote the time and enthusiasm he has to a student's "dream". Also deserving mention are Kathy Burg who typed the report and corrected my spelling, and my wife, Linda, who served as a comforting influence during my "pseudo-pregnancy" before the birth of this seminar.

FOOTNOTES

1. A rating less than 5 meant the topic should be excluded so more time could be spent on other topics.
2. This will be done on a most elementary level since each of these areas could easily take up whole courses.
3. With guidance and active participation by Dr. James Simmonds, Professor of Applied Mathematics.
4. Tests might still be appropriate as feedback to participants.
5. It is not uncommon to wait 2-1/2 hours in student health before seeing a doctor.
6. Both are published by Prentice-Hall, 1970. The sections will soon switch text.
7. In "Technology Review", Vol. 73, No. 3, January, 1971.
8. The trainer was Ed Jones from the Federal Executive Institute.
9. The human poets were Kenneth Koch, E. E. Cummings, and Richard Shelton.
10. It is interesting that this suggestion was rated lowest by those turning in questionnaires.
11. The statement really contains two issues--should technology be accelerated and should everyone have a high quality of life.
12. "The Management of Information and Knowledge", Proceedings before the Panel on Science and Technology, U. S. House of Representatives, 1970.
13. Out of a possible 16.

COURSE PRESENTATION UNITS ON SOCIAL IMPACT OF COMPUTERS

Ronald E. Anderson
University of Minnesota
Minneapolis, Minnesota 55455

The recent flood of books on "computers and society", "problems of automation" and the "organizational impact of computing" suggests a rapidly growing concern for incorporating these ideas into instructional activities. Accompanying this trend is the emergence of many new courses such as "Cybernation and Man" (Parkman, 1967) and "Society Computers and Values" (Lewis, 1970). Instead of proposing another course, this statement explores how topics of the "social implications of computers" might be integrated into existing courses. The presentation units outlined might easily fit into a wide variety of courses including those in computer science, social psychology, social organization, as well as those specifically on social applications and social implications of computers.

Underlying this approach is the assumption that on the computing side there is too little attention paid to social consequences of computer technology and on the social science side there is inadequate attention given to applying social scientific approaches to the integration of computers into society, especially man/machine relationships. Existing courses would become more interesting and relevant by the addition of these topics.

Below are nine topics which should be developed. There are obviously other ways to subdivide the problem areas and undoubtedly important topics omitted from these outlines. This is intended as a starting point rather than a rigid guide. Each topic is conceived as a "presentation unit" which might naturally fill one or two class periods, although the topic could easily take up a full course in itself or be reduced to a five-minute summary.

SOCIO-COMPUTER TOPICS

I. SOCIAL PSYCHOLOGY OF MAN-MACHINE INTERACTION

A. Perceptions and feelings toward machines, especially computers. Mueller (1969), Lee (1960).

B. Computer user roles, e.g., consumer and controller roles; psychological and social characteristics of computer users.

C. Role playing in situations of natural language dialog with computers.

D. Social psychology of depersonalization. Taviss (1970).

II. SMALL GROUP COMMUNICATION AND COMPUTERS

A. How computing affects small group interaction. Bryan (1970).

B. How small groups affect computer systems. Yershov (1965).

C. How to use computers to stimulate useful social interaction. Bryan (1970).

III. COMPUTERS IN FORMAL ORGANIZATIONS

A. Changes in Management and Vulnerability of Management. Withington (1969).

B. Impact upon alienation, job satisfaction, status, security of workers. Whisler (1970), Mueller (1969), Marcson (1970).

IV. IMPACT OF COMPUTER UTILITIES UPON COMMUNITIES

A. Information utilities. Sackman and Nie (1970).

B. Computers in the home. Martin and Norman (1970).

V. SOCIAL CONTROL IN A COMPUTERIZED SOCIETY
    A. Resistance to Computers. Myers (1970).
    B. Public versus private control of information utilities. Sackman and Nie (1970).
    C. Law enforcement. Myers (1970), Martin and Norman (1970).
    D. Military Control.

VI. SOCIAL INFORMATION SYSTEMS AND THE PRIVACY PROBLEM
    A. The Privacy Problem. Miller (1971), Myers (1970).
    B. Computer Techniques which preserve privacy. Sawyer and Schechter (1968).
    C. Urban Information Systems and Data Banks.

VII. INTERNATIONAL IMPLICATIONS OF COMPUTERS
    A. Impacts on political systems. Martin and Norman (1970).
    B. Underdeveloped Areas and Computers. Parkman (1967).

VIII. SOCIO-ECONOMIC IMPACT OF COMPUTERS
    A. Effects of automation upon occupational structure, manpower. Marcson (1970).
    B. Effects upon division of labor, nature of tasks, and the flow of cash. Martin and Norman (1970).

IX. TOWARD A PHILOSOPHY OF SOCIETY AND ITS COMPUTERS
    A. The evil and the good of technology. Ellul (1964).
    B. Philosophy for the computer utility of the near future. Sackman (1967).

Although taken as a whole these nine units would constitute an interesting course on the social implications of computers, some of the ideas should be integrated into other courses, especially a course on the social applications of computers. I currently teach a course on social science computing; the longer I teach it, the more social-impact material I squeeze in. The course would benefit greatly from a systematic presentation of Topic VI (Social Information Systems and Privacy), and Topic I (Man-Machine Interaction), as well as passing attention to Topic V (Social Control) and Topic IX (Philosophy of Computing).

ACKNOWLEDGEMENT

This work has been supported by the Center for Curriculum Studies, University of Minnesota.

REFERENCES*

Bryan, Glenn L.
1970 "Student-to-Student Interaction in Computer Time-Sharing Systems", *Computers and Automation*, (March), 18-23.

Ellul, Jacques
1964 *The Technological Society*, Alfred Knopf and Vintage Books, paper.

Lee, Robert S.
1970 "Social Attitudes and the Computer Revolution" "*Public Opinion Quarterly*, 34, 1, (Spring), pp. 53-59.

Lewis, Jerry M.
1970 "Society, Computers and Values: A Course Proposal", Unpublished Paper, Department of Sociology, Kent State University, Kent, Ohio.

Marcson, Simon (ed.)
1970 *Automation, Alienation, and Anomie*, Harper and Row.

Martin, James and Adrian R. D. Norman
1970 *The Computerized Society*, Prentice-Hall.

Miller, Arthur R.
1971 *The Assault on Privacy*, University of Michigan Press.

Mueller, Eva (et al)
1969 *Technological Advance in an Expanding Economy*, Institute for Social Research, University of Michigan.

Myers, Charles A.
1970 *Computers in Knowledge-Based Fields*, Cambridge, Mass., MIT Press.

Parkman, Ralph
1967 *Cybernation and Man: A Course Development Project*. School of Engineering, San Jose State College, San Jose, California.

Sackman, Harold
1967 *Computers, System Sciences and Evolving Society*, New York: John Wiley and Sons, Inc.

Sackman, H. and Norman Nie (eds.)
1970 *The Information Utility and Social Choice*, Montvale, New Jersey: AFIPS Press.

Sawyer, Jack and Howard Schechter
1968 "Computers, Privacy and the National Data Center", *American Psychologist*, 23, pp. 810-818.

Taviss, Irene (ed.)
1970 *The Computer Impact*, Prentice-Hall.

Yershov, A. P.
1965 "One View of Man-Machine Interaction", *Journal of the Association for Computing Machinery*, Vol. 12, No. 3 (July), pp. 315-325.

* A more extensive bibliography is available from the author at the Department of Sociology, University of Minnesota.

SOCIAL IMPLICATIONS OF COMPUTERS AT SCRIPPS COLLEGE

Lois Langland and F. W. Weingarten
Scripps College
Claremont, California

-A Discussion Summary-

A course on the topic of Computers and Society was taught twice this last year (1970-71) at Scripps College, an undergraduate, liberal arts women's college in Claremont, California.

The course was a seven week seminar titled "The Human Use of Computers" taught as one of a series of short seminars offered under the general rubric "Images of Man". The seminar, directed by a team consisting of a computer scientist and a psychologist, was primarily a reading and discussion course, although the computer scientist lectured occasionally when technical backup was necessary. A small ten-page paper and oral report was required. These papers usually involved an examination of the use of the computer in a field of interest to the author.

Because of the specialized context and short time available, the goals of the course were modest and oriented toward exploring two points; how the computer will affect man's view of himself and society, and the computer as a paradigm of technology in terms of its interaction with society. The topics which led to exploration of these points were artificial intelligence, privacy, and CAI.

Students taking the course ranged from sophomores to seniors and their computer background varied from total ignorance to heavy previous computer use. Although a few lectures were given discussing what a computer was and how it operated, no programming was taught. Programming background was not observed to have significant effect on the ability of students to handle the discussions.

SOCIAL APPLICATIONS OF COMPUTING

Thomas E. Kurtz
Dartmouth College
Hanover, New Hampshire 03755

Computing for the Non-Scientist

The applications of computing to social areas are clearly more important than applications within scientific research, even though the latter came first and supplied the raison d'etre for computing. Recognizing this fact, and further recognizing that 70 percent of its students choose majors outside the sciences, Dartmouth instituted about five years ago a course entitled "The Role of the Computer Outside the Sciences". A textbook for this course was also prepared.

The course includes very little "computer scientific" material, and teaches programming technique only as needed to illustrate the ideas and carry out the projects. The language used is BASIC, which diminishes about as much as is possible the time and effort spent learning programming.

The course opens with a two week review of the elementary BASIC the student was supposed to have learned in his first mathematics course. Then follows about four weeks devoted to the important ideas of simulation, game playing, string handling, and file systems. Each of these is accompanied by realistic non-scientific applications examples and just enough programming technique to allow the students to understand and program examples in these areas.

About mid-course each student selects a term project--a major effort involving the planning and constructing of a non-trivial program or system directed to an application of personal interest to the student. As the project progresses, continued consultation and guidance is provided to help the student pace himself, and to protect him from choosing paths that are too long or complex for the ten week term.

In evaluating the term projects, strong attention is paid to the planning or systems design, the documentation (both written and in the program), the user interface, and (most importantly) whether or not it works. The term project contributes 40-50 percent of the course grade; there is no final examination.

While the students are presumably hard at work on their projects, the lectures continue, covering special applications, usually with one topic per lecture. During this period, we might present talks on such subjects as: specific applications to business, administrative data processing, library automation, classical literature and concordances, music, and the use of special terminals and plotters.

Computers and Society

An experimental course devoted to studying the direct application of computers to social needs was offered recently. More intensive than the course discussed above, it still retained the non-physical-science character of the other course. It was conducted as a seminar, with the students divided into teams of 4 or 5 for the purpose of developing a major project. The discussions in the seminar related to the applications chosen and were not concerned directly with such computer-scientific notions as programming technique and efficiency.

In the winter of 1970-71, the eight projects were:

1. Conversion of Forrester's City Simulation Model to BASIC; study and research based on it.
2. Development of a system for matching applicants to available spaces in colleges, much like the national medical intern-matching system.
3. Design and construction of a faculty data base with suitable user interface programs. (This project brought to light some of the issues of privacy and security of data, and methods for protecting both persons and data.)
4. Development of a freshman advising project to suggest what courses to take in what terms, based on the student's interests and the college requirements.

5. Design of a freeway traffic simulation model to study the effects of controlled entry and exit. (They were forced by time to use fictitious data.)

6. Development of a system to assign housing to older residents in a nearby town according to federal priorities for Public Housing agencies. They used real data, but ran into time problems.

7. Construction of an outpatient appointment system for a medical clinic; this one came close to being actually used.

8. Building a medical data base system for patient records; a broad project but forced to use fictitious data. This project was clearly too large for the time period.

Summarizing, the faculty data base system is now in active use. The city simulation was very successful, and led to some results which were communicated to Professor Forrester. The other projects were diminished by (a) having to work with fictitious data, or (b) the severe time constraints imposed by a college term. I feel it is most important to the success of such courses to find ways in which several terms or a year can be allowed for project completion.

AN EDUCATIONAL MINICOMPUTER FOR THE LABORATORY

Charles L. Wilkins
University of Nebraska
Lincoln, Nebraska 68508

Charles E. Klopfenstein
University of Oregon
Eugene, Oregon 97403

Although technological advances in the computer science field have now made possible the extensive implementation of minicomputers for educational purposes, the most perplexing problem which faces an educator who wishes to make use of this technology is the question of how to get started. When a prospective user faces the bewildering variety of computers and competitive claims about their capabilities, he may doubt the possibility of ever correctly evaluating and selecting the computer best suited for his application. One purpose of this paper is to outline some procedures whereby the beginner may reduce these problems to a workable minimum and, at the same time, make possible some hope of at least avoiding disastrous mistakes. For the purposes of our discussion, we will assume the need for some type of digital equipment has already been established and direct ourselves to how one might proceed from there. Where does one begin? In the following paragraphs we will trace through a typical procedure which might be followed by a potential user. Once we have done that, we will discuss the implementation of a system which evolved in this way.

The Preliminary Stages

In the first place, it is readily obvious that if one is to evaluate the technical claims made by computer manufacturers, it is necessary that he have sufficient background and expertise to place the claims in the framework of his own situation. It is still true, unfortunately, that caveat emptor is the rule in the intensely competitive computer marketplace. In order to be able to obey this rule, it is incumbent upon the purchaser to know at least enough about the equipment he intends to buy so that he may understand its limitations as well as its capabilities. Quite naturally, the manufacturers stress the strong points of their particular machines and "gloss over" the weak points or limitations. In other words, if these are to be found, they must be found by the customer, for the manufacturer is unlikely to point them out to you. The most pressing question at the outset, then, is how to obtain the necessary information in order to make the best choice for your own particular needs. There are a number of ways that this might be done. If funds are available, the most straightforward way is simply to add to your faculty or staff someone with the required expertise. In these days of reduced budgets, this solution, while optimum from an efficiency standpoint, may not be practical from an economic standpoint. It is clear that if this could be done the evaluation phase of computer implementation would be a relatively trivial problem. A second approach is for a person in the department hoping to use the equipment to become trained to the point that he can at least evaluate the claims, even though he may not be an expert. This is probably the solution most academic departments will choose. The reasons for this are twofold. First, it is much easier to train someone already on your payroll than it is to hire an additional staff member. Second, and perhaps more important, if such equipment is to be successfully used in your department, it will require the presence of someone who is reasonably knowledgeable. For, if there is no one of this sort, it will be impossible to make much use of the equipment and the limitations would be such that it would be difficult, or impossible, to adequately train students to use it. We therefore consider the presence of a suitably trained faculty member as essential to the successful use of digital computers in the undergraduate curriculum.

The next decision to be made, once it has been decided to train a present staff member, is how this training is to take place. There are various alternatives available, some of which are more satisfactory than others. One way a person may be trained is to spend two to three years talking to minicomputer manufacturers, reading their literature, examining their machines, learning how to program computers, and visiting users who may have the machines in operation. This is, in our opinion, an unsatisfactory means of training a faculty member. For one thing, it is a relatively inefficient use of his time and it is entirely possible that the difficulty of acquiring the necessary knowledge in this way may extend the time required from the minimum two or three years we have just mentioned to as much as five or six years. In the meantime, the equipment will have changed so much that the basic plans for implementation may be entirely different. In any case, this approach implies that the person in question has the necessary background to allow him to learn what he needs to know in this way. While this may occasionally be true, it is more likely not to be the case. The second, and, we believe, more effective means of training, is for the staff member to attend various short courses, ranging in length from a day or two to several weeks in order to gain the required knowledge. In general, we do not recommend a user expect to learn enough in a course of one or two days, for the purpose we are discussing. Rather, intensive short courses (e.g. Dr. Sam Perone's course for chemists at Purdue University) lasting from one to three weeks and operating from eight to ten or twelve hours per day are the method of choice if time limitations are significant. Alternately, where such courses exist, a one or two-semester course at a college or university should also adequately satisfy the need for fundamental information. This sort of training course should not be confused with the much more common courses in FORTRAN or other programming languages offered by most colleges and universities which have computer facilities. Such courses are not aimed at the understanding of the equipment or its operation, but rather at its use and, as such, are of relatively little value in enabling a person to evaluate and select digital computing equipment.

Defining the Needs

Once the preliminary phase has been completed and there is now someone available in your department who has basic knowledge of minicomputers, it is then possible to begin defining your particular needs. Note, we did not suggest you try to define your needs before the acquisition of trained staff personnel. The reason for this should be obvious: the basic information necessary to define the needs is not usually clear to a person until he has some knowledge of the capabilities of the equipment with which he contemplates satisfying those needs. In particular, it is necessary to understand what experimental parameters are of importance. We believe this can only be done once some preliminary training has been completed. First, it is important to examine your curriculum and to determine where effective use of the laboratory computer might be made. If the computer is simply to be used as a sophisticated toy to momentarily engage the student's attention, it would probably be wasted. It is highly important that if the computer is used, it be used in such a way as to take advantage of its capabilities rather than to simply do the same old things in a more expensive fashion. Accordingly, a careful examination of the curriculum is in order at this point. In general, computational problems of any complexity will be far better handled by a large central campus computer (perhaps with remote terminals attached) and, if this is your principal or only application, it is entirely possible that a minicomputer will not be suitable for you. On the other hand, minicomputers are particularly good at acquiring data from experiments and controlling experimental parameters. If this is your area of application, it is more than likely that the central campus computer, if any, will be completely ineffectual for your purposes. In this situation, a minicomputer is the indicated choice. So, it is clear that how the computer will fit into your curriculum is an important variable in the decision about what equipment is required.

Second, one must consider the students who are to use the equipment. What are their backgrounds in computer use? Are they likely to have had prior experience with computers? Are they likely to have had previous courses in programming? How many students are there? How many of them will make much use of the equipment once it is acquired? Are they already making use of some computational equipment? All of these are questions which should be answered and will be important in the definition of the needs. Third, if in the analysis of the first two factors it has been determined that a laboratory computer is the requirement, what will be the specific sorts of applications in the laboratory? Will data from relatively slow experiments be acquired? Will control be necessary? Will real-time data reduction be necessary, or can this be done off-line? What sorts of computations will be required? Is there really a need for a computer in your laboratory? Can some sort of a data transmission or logging device serve your need (e.g. PAR Instruments markets a device for acquiring data from experiments and sending it via telephone lines to a remote computer for processing; numerous logging devices, ranging from paper tape to punched cards and magnetic tape units are available, if data logging is all that is required)? These are essential questions which must be answered in the planning phase of implementation of laboratory computers in the curriculum.

Finally, once this preliminary evaluation has been completed, it can be very much to your advantage to consult with experts about what might be your course of action in light of the needs you have defined. If this early training and evaluation has been done conscientiously, it will now be possible for an expert to give you some reasonable alternatives and possible approaches to the meeting of your needs. So, at this point, it would be of great advantage to consult one or two of those who have worked in this area for several years, in order to take advantage of their knowledge and experience. Consultation of an expert can save you many costly mistakes.

Now you will be ready to begin the most difficult part of the whole process. Presumably, you will know what type of equipment is required and what its functional characteristics must be. Hence, you are in a position to establish a realistic working budget. Foremost in your consideration at this point will be the decision about whether to purchase or lease the required equipment. There are advantages and disadvantages to both plans. It is, of course, clear that if you purchase the equipment, you will be required to provide the maintenance. This may either be done under a maintenance contract or with your own personnel (if you have such trained technical personnel). These are matters which should be important parts of establishing any such equipment budget. For if the equipment is not working, it is of little use to you. Immediately after this comes the part of the task we just referred to as the most difficult component of the whole process of implementing the digital computer in the undergraduate laboratory curriculum. Find the funds! On this particular point we claim no special knowledge, but suggest that the administration of your college or university must be convinced there are significant and essential educational advantages in the use of a laboratory minicomputer. This is a case which can be made, especially in view of the widespread implementation of such equipment in the technological world which graduating science students will enter. Obviously those trained in the use of this equipment will be in a position of advantage in the job market. While it is sometimes unfashionable to consider such mundane matters, we think that this, more than many other things, may satisfy some of the current demand for curricular "relevance".

The Hardware Evaluation Phase

It is only now, after the preliminary work described above has been completed, that you are ready to begin hardware evaluation. It is particularly important to realize that if you try to evaluate hardware before taking the steps described above, you will be under considerable pressure from the salesmen you meet to formulate your needs and requirements in terms of the capabilities of the hardware they have to offer. While this is certainly desirable from the standpoint of the computer manufacturer, it is necessarily less than desirable from the viewpoint of the end user. The computer market is sufficiently competitive today that there is no reason not to have a system which closely approximates your requirements. Accordingly, the first step to take as you prepare to evaluate digital computer hardware is to formulate a detailed set of functional specifications to submit to manufacturers. The advantage of having such a set of specifications in writing is that it eliminates all misunderstanding at the outset and provides a convenient reference point, should later disagreements about specifications arise. Naturally, it is impossible to write a reasonable set of specifications unless you are sufficiently familiar with first, your own needs, and second, what is possible within the present state-of-the-art. This simply emphasizes that it is necessary to be somewhat knowledgeable in order to adequately evaluate the hardware.

Does it work? That, of course, is the important question when you reach the stage of evaluation where one, two, or three computer systems appear to be most nearly adequate for your purpose. The large computer manufacturers have a long history of making promises which they can't keep at the time they make them. These promises represent, instead, informed estimates of what they might be able to do in the future. This is standard practice in order to meet competition. It is rare that all manufacturers simultaneously arrive at advances. Therefore, in order to stay in business, those who have not yet reached a particular point (but are working on it) will often claim that the accomplishment is merely a week, a month, or six months in the future. They may sincerely believe this, but paraphrasing the poet, "...the best-lvid plans of mice and men often go astray...". It is therefore a reasonable request of any potential seller that he allow you to examine operation systems doing the things you plan to do. If he is reluctant to do this, or if he refuses to do it, you should seriously think about eliminating the manufacturer in question from further consideration. It is probably particularly important that you see such demonstrations in the "real" world, rather than at the manufacturer's plant. It is quite appropriate, if possible, to visit other users who have already implemented the equipment in their own application. It is from these users that you are likely to get the most forthright comments and reliably objective information. Even when talking to users who have implemented certain equipment you may be considering, remember that these people have already committed funds and made the decision that this equipment is the best. It may very well be, for their purposes. Obviously though, they will be likely to be enthusiastic about it unless it isn't working at all. It is unlikely that the manufacturer will direct you to any people in the latter category, so you should weigh this factor in considering the comments of users you may talk to. In other words, you must pay a great deal of attention to the minor criticisms they may make, for these are more likely to be of significance to you in evaluation than the more enthusiastic endorsements of the equipment, which are far more likely to be offered. Whether or not you get the required information will be largely a function of the questions you ask. While it is probably safe to say that, if you've carried out the steps we have outlined above, any of the two or three manufacturers you might settle on at this point would be able to satisfy your needs, to a greater or lesser degree, it is important to get the most capability for the money available and this sometimes hinges on more subtle factors.

What about demonstrations at the manufacturer's plant? One of the things you may be told, as a potential buyer or leaser, is that the equipment is so new and so advanced that there has not been time to install it in users' laboratories or institutions. This may be true and, if it is, the only possibility for demonstrations may be at the manufacturer's plant or at manufacturer's applications laboratories. If even this possibility does not exist, in most cases it would be wise to assume the equipment is not yet quite ready to market and you should act accordingly. Remember, when you do see a demonstration on the manufacturer's terms, it will be under the best of all possible conditions. The service and maintenance such equipment receives is almost certain to be far more extensive than that which would be available at your site. If something should go wrong, it will probably be quickly rectified and its importance minimzed. However, "minor" problems which are quickly rectified at the manufacturer's plant may take considerably more time and effort at your institution. This should be considered.

Software Evaluation

Once more, we cannot stress too strongly that it is necessary when evaluating software or hardware to consider your application of the equipment. By the time you begin software evaluation, you should, as we have mentioned, have narrowed your possible choices to a few computer systems. Although for purposes of discussion we have separated hardware and software evaluation, in actual practice it is necessary to perform these simultaneously. In order to be able to take advantage of the hardware features, necessary software must be available. Conversely, in order to make the most use of the software, it may be necessary to have a certain hardwrre configuration. This is a point which should be most carefully considered. As before, it is quite beneficial, when evaluating software claims, to talk to other users who have implemented the system in question under the same or similar conditions as those you visualize in your application. There are a number of questions to ask at this time and we will try to outline a few of them. It is probable that in any particular situation there will be additional questions of interest and the outline we present here is only intended to enumerate a few of such questions and not to be comprehensive.

With almost all digital computer systems you are likely to consider, a mnemonic assember will be provided. Some of the important questions to be asked about this are the following:

1) Does the assember produce relocatable object programs?
2) Is it a single-pass or a multi-pass assembler?
3) How many mnemonic instructions are defined?
4) What instructions are available for controlling the operation of the assembler?

Another important question to ask with regard to software is whether a suitable symbolic editor program is available. Programs of this type are extremely useful for the correction of programs written in mnemonic code and are essential to efficient use of any assembler. Another sort of software support also important is a program for use in debugging applications. It is often desirable to be able to display and modify the contents of memory locations and to be able in some way to monitor the operation of programs written in assembly language. Therefore a powerful debugging program is a highly desirable type of software which, if not supplied by the manufacturer, may well have to be developed by the user. Of interest to most people in an educational environment is the question of whether high-level languages are available for use with the system. In other words, will it be possible to use such languages as BASIC, FORTRAN, FOCAL, and the like? If it is possible, what are the limitations? Many times, although the languages are advertised as part of the software provided by the manufacturer, closer examination shows that what he will call FORTRAN (for example) is actually a limited subset of the language most commonly familiar to potential users. So, it is necessary to inquire into the details of the high-level languages which are provided, if it is claimed they may be used. Also, it is necessary to investigate the hardware requirements for meaningful use of the languages. For instance, it is possible to buy 4,096-word computers which have BASIC compilers available. However, the memory requirement of the compiler is such that use of more than the smallest programs is imposOible. While this may be satisfactory for some purposes, it is likely to be a significant limitation for most applications. It is therefore much more common that at least 8-K core memory is required if the user wishes to implement BASIC. It can be seen from this example that caution is necessary when evaluating software support. Finally, are there assemblers and simulators available (and will they be provided) to allow a larger computer such as the IBM-360, or similar computers, to assemble code and simulate its operation for the laboratory computer you may be considering? Software such as this can be of significant use in freeing the laboratory computer for use with experiments and permitting many more students to make use of it than might otherw'se be possible. Therefore such software can be of significant importance and should certainly be used if available.

The Big Step

Having completed all the steps outlined above, now you are ready for the big step! You are ready to select the system and acquire it. It is when you reach this point that you must remember one important thing which has been true in the past and will continue to be true. Whatever choice you make will be the wrong one! That seems, at first glance, like a startling assertion to make after devoting a major part of the paper to describing a step-by-step process for making the right choice. It is, frankly, one of those attention-getting devices which are often used to bring out a true point which has been embodied in the statement. The advances in computer technology are still proceeding at such a rate that whatever you buy today will, in all probability, by the time it is delivered be "obsolete". It will be obsolete in the sense that it will not represent the highest or most advanced state-of-the-art at the time you receive it, even though it may have represented that at the time you decided to purchase it. This does not mean that the equipment you select will not serve your purpose adequately. It does mean that you may encounter some people (often salesmen) sometime between the time you make your choice and the time it is delivered who will claim that you ought to reconsider since now the equipment you have selected is obsolete. This is standard strategy on the part of the salesmen for, until the equipment is actually received, the competing salesmen who represent companies offering equipment you did not choose will continue to try to sell you their equipment. The longer they can delay you in your final choice, the more chance they have to switch you to what they are selling. Do not fall into this trap, for you will never be able to receive equipment which is not somewhat outdated by the time it arrives. If you are going to wait for the perfect solution to your problem, you will never acquire any equipment. It is not likely the perfect solution to any problem in laboratory computing is, or will be, available in the near future.

Once you have chosen your system, acquired it, have it set up and operating, what then? Well, in an educational institution, the technique most teachers have found extremly useful is to select some highly motivated interested competent students and to show them how to use the computer. Let them get their hands on the equipment and actually make use of it. If you find students who fit this description, it is usually a matter of only a short time before they begin to conceive and execute projects which illustrate the ingenuity and ability of students in going beyond the basis we provide them in formal courses. Especially if education in the uses of laboratory computers is your aim, such students may prove, in the long run, to be the best teachers of their fellow students and will provide much worthwhile encouragement to other students just beginning their studies. If the beginning students see more advanced students doing things which to them, at their present state of knowledge, seem difficult--it brings home to them the fact that this is indeed an area of excitement, of interest, and of potential (with a little hard work on their part).

In conclusion, it has been the purpose of this part of the paper to attempt to outline a broad general approach to getting started in the use of an educational minicomputer in the laboratory. The approach we have outlined is one which we personally have found to be highly successful, and one which we believe will enable the beginning user to avoid the major pitfalls inherent in entering a somewhat new field. While, of necessity, we have omitted a great number of details that might have been considered in such a description, we believe that someone following this outline as he progresses from an interest in bringing a minicomputer into his laboratory to actual implementation will not go far astray. The steps are fundamentally simple and can be summarized in a short sentence or two. First, acquire the training necessary to understand the needs and problems. Second, define your needs, establish a realistic working budget, and find the funds. Third, evaluate the hardware and software offered and find that particular combination best suited for your requirements. Finally, select the system and acquire it, then put it into operation. Progressing through this sequence of steps in the design of a general-purpose laboratory teaching computer system has led the authors to one such system which fills the specifications contained in the Appendix. A photograph of that system is included (Figure 1). For approximately $35,000, the system pictured has been put together and used for training students in the use of a laboratory computer for data acquisition and control applications. Numerous other uses of this particular system have also been explored. We believe this is proof that, following the guidelines we have outlined above, it is possible to make a reasonably good choice of a computer system which satisfies a particular set of educational and economic criteria. In the next part of the paper the teaching applications of the system we chose will be discussed.

Approach to Course Development

By the time funds have been obtained and the computer to be used is on order, as we've mentioned above, at least one member of the staff should have become expert in the languages and capabilities of the particular computer chosen through one of the methods described earlier. His first task after the computer has been delivered is to familiarize his co-workers, his teaching assistants, and a selected group of students with the machine and with digital logic. It has been our experience that this step is necessary before laboratory computer experiments are incorporated into existing courses.

A. Teaching Digital Logic

We have found that undergraduate students in Chemistry and other fields can easily learn the principles of digital logic and integrated circuitry. The technique used was to first deliver two lectures on Boolean Algebra, and then two lectures on the functions of various logic elements. While lectures discussing various functions continue, we allow the students to perform experiments using a "computer laboratory" of the type available from the Digital Equipment Corporation. The device is sold complete with the student workbook and the teacher's manual. We have examined similar products available from other manufacturers (Heath, El Instruments, and others). Most of these also seem adequate for educational purposes. However, the most flexible approach after the initial experience with the "computer lab" has been to allow students to wire circuits using a simple breadboard device. The breadboard we chose for this purpose is available from API Corporation, and will hold dozens of 14, 16, or 24 pin integrated circuits. Several Heathkit IP-19 power supply kits were purchased, along with the necess8ry wire and tools for implementation.

B. Teaching Assembly Language

The assembly language lectures at both Nebraska and Oregon start with a thorough description of the hardware of the 620/i computer. This accomplishes the task of both familiarizing the students with the architecture of small machines and, at the same time, drives home the simplicity of design of such machines. Wherever possible, we stress the trade-off made between simplicity for low cost, and performance. The background the students gain in this fashion eases their acquisition of a working knowledge of machine language. Particularly, the bits of an operation code or an address modifier in machine language are instantly understood after such a discussion rather than merely memorized.

We realized from the outset that if the training of large numbers of students was not to be prohibitively expensive, we must devise a suitable means to allow as many students as possible to use the computer hardware. While we had been waiting for the hardware to be delivered, we modified an IBM-360 assembler to be more effective as a teaching tool. This allowed all the students access to an assembler in the pre-hardware delivery phase and, after hardware was delivered, allowed the students the possibility of punching their assigned programs on standard IBM cards and submitting them to our IBM-360 computers for assembly in the same way they would submit a FORTRAN job. We have found this is a particularly important development, since in the early phases of the project, it became apparent that as much as 80 to 90% of the teaching computer time was devoted to program debugging, if the 360 assembler system were not available. The object code produced by the assembly on the 360 may be written to a master magnetic tape, and if a 620/i with paper tape output facilities in addition to magnetic tape facilities is available, may be transferred to paper tape. Alternately, if paper tape facilities are available on the 360, the programs could be directly punched on this output device. After a successful run, the student may load his program and by methods described later, determine that it is operating properly before turning in his work as a completed assignment. It is during these latter steps that the student gains his "hands-on" experience with the hardware. This approach has been highly successful in teaching assembly language to both experienced and inexperienced students. Furthermore, even in situations where it is impossible to directly transfer the IBM-360 assembled program to the 620/i, the student may still be spared numerous slow and tedious assemblies during early stages of program development, and need only use the 620/i hardware at an advanced stage in the development of his program. At Nebraska, where we had no means of directly transferring our 360 output to the 620/i, we found this program was still almost as useful (of course, it would have been preferable to be able to transfer the output directly).

C. High Level Language Experimentation

Experience at Oregon has demonstrated that it is impractical to expect students who select computer experiments as part of other courses to learn assembly languages. We have explored the use of three high-level languages to help these students progress, as well as developed a set of teaching cards containing experimental circuits. The student, then, is not required to learn the intricacies of either digital logic or machine language, and further has the advantage of ready transference of what he learns to other computer systems. The languages examined were FORTRAN, BASIC and CLASS. All three were found to be useful for educational purposes. FORTRAN and BASIC require 8K of core storage and have the advantage that most students already have had some contact with one or the other languages, and CLASS has the advantage that it requires only 4K of storage for operation.

Some of the students participating in the courses have neither the time nor inclination to learn enough digital logic to perform experiments. To provide experiments for these students, we designed and implemented a special card and socket arrangement. The standard socket brings out all of the analog and digital functions required to perform most experiments. Students merely select the card that contains the experiment, and insert it in the socket. The process is so fast that a single student has time to debug several experiments at one sitting. Experiments include a voltage controlled oscillator, a comparator, and a function generator.

On the basis of our feasibility studies, we have shown that it is clearly superior for beginning students to make use of a high level language for their initial contact with the computer system It might be argued that FORTRAN should be the language of choice for this purpose, since many of the students who are in the sciences already have had some contact, perhaps even a formal course, in the language. For the language to be useful in a laboratory environment, drivers must be provided to support all the laboratory peripherals. We have done this, and found this approach to be quite satisfactory and highly successful, proving that experiments can be performed by students who may have never seen a small computer before, and who have only a relatively short time to devote to learning real-time computing techniques. The language BASIC has just recently become available for use on the Varian 620/i. In contrast to the language FORTRAN, BASIC runs interpretatively. The students can therefore write, edit, and debug programs all in one sitting. For example, in one typical experiment the student is required to prepare a computer program that acquires data from a discharging capacitor, and from these data calculate the capacitance versus voltage for the device used. He also plots voltage versus time and capacitance versus voltage on both an X-Y recorder and an oscilloscope. Towards the end of this term, the capacitor will be replaced by a chemical reaction followed by monitoring pH or conductance. The BASIC programs to perform the data acqusdition are usually less than 20 lines long and so don't take long to enter into the computer, nor do they use up much of the minicomputer's memory. The disadvantage of BASIC, which for educational purposes is no disadvantage at all, is that it is roughly fifteen times slower than FORTRAN. We have provided standard BASIC with the same functions and subroutines as we did for FORTRAN. Although it is too early to make a final judgment, we so far totally concur with the results of Professor Sam Perone, who has found that BASIC with real-time modifications can be a very powerful teaching tool.

D. Experiments in Simulation

We have found that the pure computational and graphic display capabilities inherent in the laboratory system can be used to great advantage in enhancing courses that teach the theory and use of Electron Spin Resonance, Nuclear Magnetic Resonance, and Kinetics. All but the Nuclear Magnetic Resonance calculations require an 8K system, and even there we have developed an 8K version that is much more powerful than the older 4K program. The Resonance simulators display results on either a 611 storage oscilloscope, or on an X-Y recorder, and allow in-core comparison of synthetic and real data. The prime advantage offered by using the laboratory computer for these simulations rather than a larger batch computer is that the students performing the simulations gain a much greater understanding of how the parameters of the

Hamiltonian effect the appearance of the spectrum, since it is possible to perform the calculations one directly after the other. The kinetics simulation program is written in BASIC and allows 9 reactants, 9 products, and stoichiometry of from 1 to 9 in each component. Up to 20 reactions can be simultaneously accommodated, and the concentration of each component is displayed on the storage oscilloscope or on an X-Y recorder for some specified time period. Again, the student gains an understanding of the dynamics of kinetics that would be difficult to obtain in any other fashion. The programs have been used to supplement instruction in Physical Chemistry Laboratory, and Organic Qualitative Analysis.

Conclusions

To date, approximately 40-50 students have participated in the combined Nebraska and Oregon projects. The students who have participated have done so purely on a voluntary basis, and so for the most part, they were interested in the subject before enrolling in the course. In the winter term of 1971, we incorporated some experiments into the standard Physical Chemistry Laboratory curriculum, a few of which all students perform and a few of which are optional. They include both simulations and data acquisition tasks. The results to the date of this writing are encouraging. Details concerning the students point-of-view will be polled at the end of the term through a course evaluation survey. It is obvious, however, that instruction in laboratory computing is not only timely, but that students with minimal background are as capable of successfully completing the courses as the more experienced students. Further, we have found that these new techniques and procedures have stimulated the interest of some students previously doing only what had been required into performing experiments on their own.

A full report of our experiences during the first two years of the project will be prepared during the summer of 1971, and it, along with the computer programs developed, will be available on request.

Acknowledgement

The authors would like to thank the National Science Foundation for their support through Grants No. GJ-441 and GJ-393.

APPENDIX I - Teaching Computer Specifications

(Appendix omitted, contact author for further information.)

THE INTRODUCTION OF DIGITAL COMPUTER TECHNOLOGY INTO THE UNDERGRADUATE CHEMISTRY LABORATORY

S. P. Perone, F. E. Lytle, and J. F. Eagleston
Purdue University
Lafayette, Indiana

ABSTRACT

This paper describes an educational program designed to introduce computerized instrumentation into the undergraduate analytical chemistry laboratory. The approach involves using a high-level conversational language--BASIC--with modifications to allow on-line data acquisition and experimental control operations to be executed. The characteristics of the modified language and the laboratory instrumentation are described here. The computerized approach to laboratory experimentation does not replace existing experiments, but rather augments these experiments by allowing more rigorous data analysis and demonstrating the capability for interaction between computer and experiment.

Current trends in the chemical industry throughout the country indicate that the digital computer will soon become an indispensable part of chemical experimentation (1, 2, 3). It should be clear, then, that the chemist of the near future will be required to understand and use properly digital instrumentation and digital computers in the laboratory. Furthermore, educators at colleges and universities will need to develop undergraduate curricula providing the required background in computerized chemical instrumentation. In fact, the demand for this kind of training by practicing scientists has prompted the development of a summer short course here at Purdue for college teachers and others in chemistry and the biomedical sciences (4).

This paper describes the first steps in a long-range program involving the introduction of on-line computer applications in the undergraduate chemistry laboratory at Purdue. The introduction has been made as part of an introductory analytical chemistry course dealing with quantitative chemical measurements (junior level at Purdue). A primary objective was that none of the chemical and instrumental principles currently developed in the course be eliminated, but rather that these be augmented by the incorporation of the on-line computer into many of the laboratory experiments. The manner in which such a program may be made feasible is described below.

APPROACH

It was our intention to introduce computer technology into the undergraduate chemistry laboratory without slighting the chemistry content. The approach has been to continue to include the same basic set of laboratory experiments--except that, eventually, many will be designed to require on-line communication with the digital computer. The student is required to understand and execute the laboratory experiment as usual--but, in addition, he is required to design the appropriate programming and communication elements for optimum interaction between the computer, experiment, and experimentalist. Thus, we envision the student's achievement to be one of *experimental design*, and the *proper utilization* of computing equipment in the solution of measurement problems in chemistry. Specific aspects emphasized include: the general-purpose nature of the digital computer (*i.e.*, how it can be applied to a large variety of experimental problems with modifications in programming); the computer as a control element in experimentation; the rapid response of the computer, allowing real-time interaction with experiments--with reaction times several orders of magnitude faster than the human operator; and the computer as an integral part of the chemical instrumentation.

To accomplish the above mentioned objectives, it was necessary to recognize, first of all, what could *not* be expected within the scope of this course: (1) we could not educate our students in the intricacies of machine-language programming, and (2) we could not provide them with the electronics background requisite for the sophisticated design and construction of interfacing between computer and experiment.

What *could* be done is to take advantage of students' previous exposure to off-line computer programming (with FORTRAN usually), which many, but not all students, have obtained before the junior year at Purdue. We could also take advantage of their exposure to the fundamentals of electrical measurements in Physics courses to instill them with the most elementary concepts of amplification, response, and noise. We could expect to provide them with the essential principles of digital logic required for interface *design*.

The most important requirement for the success of the program, however, was the development of two items: (1) a modified high-level programming language, which includes versatile data acquisition and experimental control subroutines with conversational mode calling sequences. (2) A general-purpose hardware package for interface design. This apparatus would provide for patchboard incorporation of those digital and analog modules required to complete the interface between computer and experiment. The general-purpose interface package requires only that the student be able to lay out the basic logic design, timing sequence, and the analog amplification or attenuation required. He can then select appropriate pre-packaged plug-in units to implement his design. He does not have to tend to such details as level conversion, logic conversion, noise rejection, and other subtle design parameters. Yet, he will accomplish the most important part of the interface development--logic design.

PURDUE REAL-TIME BASIC SYSTEM

The high-level programming language chosen for development in this work was the BASIC language (5). There were several reasons for selecting this language: (1) It is easy to learn, generally requiring the order of half a day exposure to develop a good working knowledge. (2) It is an algebraically-oriented conversational language. (3) It is interactive. That is, the compiler is interpretive, and therefore compiles and executes programs line-by-line. This allows programs to be entered, executed, and edited on-line through a teletype terminal. It also provides for immediate turn-around and rapid error diagnostics. (4) BASIC is rapidly becoming a universally acceptable language. (5) BASIC is currently readily available at commercial time-share terminals. This allows the convenient and economical learning of the language by large numbers of students for later on-line experimental applications.

It should be noted that other languages (e.g., FOCAL*) are available which have comparably desirable characteristics. Even language such as FORTRAN or ALGOL which are generally considered strictly for off-line applications may be appropriate, particularly for systems with "load-and-go" capability.

The software system developed at Purdue will be referred to as "Purdue Real-Time Basic," (PRTB). The software includesc fundamentally, the BASIC compiler available from Hewlett-Packard Co., which manufactured the computers used in the undergraduate laboratory program (see Experimental Section). The modifications generated here at Purdue have involved the development of a series of machine-language subroutines which are directly callable from the BASIC software, and which are designed to communicate in a variety of ways with experimental systems. The PRTB data acquisition and control software are described below.

General Description of PRTB System

Figure 1 illustrates the general nature of an on-line computer system. On the one hand is located the laboratory experiment, the data from which the student desires to present to the computer for data processing. On the other hand is the digital computer itself, which is capable of high-speed programmed computational and control functions. In order to operate the communication link between the computer and the experimental system an electronic interface must be established. This interface accomplishes the functions of translation (analog-to-digital, digital-to-analog, decoding, logic conversions, etc.), timing, synchronization, and logical control. The programming language must take into account the nature of the electronic interfacing; and the interfacing must be designed with the fundamental characteristics and capabilities of the programming language in mind. The following section describes the subroutines developed for on-line experimentation within the PRTB system.

Experimental Input/Output Subroutines for PRTB

Data Acquisition. Two different types of data acquisition subroutines are available within PRTB. They both work in conjunction with an external clock-controlled analog-to-digital converter (ADC). One is a subroutine (SB3) which should be called by the user's program whenever the computer should be waiting for the next data point to become available. When the next datum is digitized, SB3 takes the data point from the data acquisition device (ADC), converts the datum to floating point format, and stores it in the appropriate memory location for subsequent reference by the BASIC program. When SB3 is exited, the computer returns to the next statement in the BASIC program for execution. The new datum can be operated on, but, before the next data point can be acquired, SB3 must be called again.

A second type of data acquisition subroutine (SB7) is one which allows for the acquisition of a complete block of data before exit. The external clock is started and synchronized with data acquisition within the subroutine. The basic difference between SB7 and SB3 is that SB3 is called to acquire one data point at a time and therefore allows for program statements to be executed during the time between acquired data points. Thus, an experimenter could devise a program which could process experimental data while the experiment was in progress and data were being acquired. This is referred to as "real-time" data processing. However, because the computer program must involve the relatively inefficient execution of BASIC statements between data points, there is a more severe limit on the speed with which data can be acquired without the computer getting out of synchronization with the experimental timing. In fact, SB3 is designed to detect when the computer has gotten out of synchronization; an error message will be typed, and the computer will halt. The user will then have to revise his program to require less real-time processing. With SB7, on the other hand, a complete block of data points are acquired before the subroutine is exited and control returned to BASIC. Therefore, the timing is limited by the efficiency of the machine-language programming developed within SB7. For our computer systems and software, it is possible to acquire data at rates as great as 20 KHz with SB7, but no real-time data processing is possible.

The limiting data rate when using SB3 is about 50 Hz, with a minimum of real-time data-handling. If several BASIC computations are to be executed between data points when using SB3, the limiting data acquisition rate may be the order of 1 to 10 Hz. However, this is generally more than adequate for most experiments in the chemistry laboratory.

Experimental Control and Logic. In addition to data acquisition, there are several other ways *in which the computer can communicate* with the experimental system. One of these is through a digital-to-analog converter (DAC), where the digital output of the computer is converted by the DAC to an equivalent analog voltage level. One of the PRTB subroutines (SB9) provides the capability for driving the DAC, the output of which can be connected to external experiments. It is possible to generate a continuous voltage waveform output from the DAC by mathematical generation within BASIC of the discrete points making up the waveform and transmitting these through SB9 to the DAC in a repetitive fashion synchronized with an external clock.

Subroutines SB8 and SB11 allow the programmer to utilize specific output bits on a selected I/O channel to control external devices. There are 16 output bits available for this function. With SB8, the user can select which bit he wants to set by specification of the bit number. The setting of one of these output bits causes a corresponding binary voltage level change at the specified output terminal, and this can be used to close or open switches, start or stop experimental events, light indicator lamps, *etc*. SB11 allows the programmer to output a 16-bit binary voltage *pattern* with any *simultaneous combination* of 1's and 0's he chooses. This binary pattern is *selected* by including in the subroutine call the decimal equivalent of the binary number to be generated. Thus, more than one event can be controlled simultaneously.

SB12 and SB13 provide digital *input* information for program "sensing" of external situations. Thus, 16 binary *voltage input* terminals are available to the user, the status of which are acquired by either subroutine. SB12 transmits to the BASIC program the numerical equivalent of the input 16-bit binary voltage pattern. Because the status of these bits can be set by external events, the computer could use this information to make appropriate changes in the data processing or control programming.

SB13 is similar to SB12, except that only the status of a single specified input bit is determined. This subroutine is useful to the student because he need only specify a bit number in the call statement to check the status of a bit.

Timing and Synchronization. The most fundamental operation which must be accomplished by the *PRTB system is the generation* of a *time base* for all experimental functions. This is accomplished by the incorporation of a *fixed frequency* crystal clock (10 MHz) into the interface hardware. Also included is electronic count-down logic to scale the output clock pulses down to a usable frequency range for chemical experimentation (100 KHz to 0.01 Hz). The countdown logic can be modified under program control to select any frequency within this which can be generated by a decade-and/or a 1,2,5-countdown sequence. The programmed clock output pulse train is then available to control the timing on the ADC, DAC, or any other external hardware. Also available simultaneously are synchronous clock pulses representing frequencies at the various stages of countdown. (These details are described in the Experimental Section.)

The clock is controlled by the PRTB software through subroutines SB1, SB2, and SB10. SB1 is the initialization subroutine. Through it the programmer specifies the clock frequency, within the limits outlined above. (SB1 is also used to specify the symbol assigned to the variable which will take on the values of the digital data acquired in SB3 or SB7.)

SB2 is called when the computer program decides that it is time to enable the clock. That is, an ENCODE bit is set to a "1" state on the data acquisition channel. This bit can be connected externally with a patchcord to the ENABLE terminal on the clock module (see Figure 2-A). Alternatively, the ENCODE bit may be brought to some external logic, the output of which will set the ENABLE clock input when other external events have occurred--like the start of the experiment. One possible external logic configuration is shown in Figure 2-B, where the ENCODE bit conditions one input of an AND gate. The other gate input is conditioned "TRUE" when the experiment has been initiated. The AND gate output will then enable the clock. At that point the countdown logic will be enabled and programmed clock pulses will begin to appear at the available outputs.

SB10 is called to disable the clock when the time base is no longer required. It simply "clears" the ENCODE bit on the data acquisition channel which has been used externally to enable the clock.

Another type of communication required between the experiment and computer is associated with synchronization between computer operations and external events. For example, if the computer has completed all preliminary program execution required before being able to accept experimental data, the user may choose to receive an output from the computer which may not only tell the experiment to start, but may also initiate external events associated with the experimental system. One subroutine available for this kind of operation is SB5. SB5 causes the output of an ENCODE bit on a selected I/O channel to change state (go to the "TRUE" level) when SB5 is called. Then the computer waits within SB5 for an external event to occur which will cause a FLAG bit on the same I/O channel to change to a TRUE state. The subroutine detects this event and then allows the next sequential program statement to be executed. This next statement might conceivably be a call to SB2 which initiates data acquisition.

A simpler, but less precise, means of communication with the computer for the purpose of synchronization is provided by SB4. When this subroutine is entered the computer simply waits until the operator flips a toggle switch corresponding to bit 0 on the computer console switch register. The computer then exits SB4 and executes the next program statement--which might be to call SB2 and start data acquisition.

Finally, a subroutine (SB6) is available which allows general-purpose timing functions. This routine simply waits for the FLAG bit to be set on the data acquisition channel, clears the FLAG, exits the subroutine, and the next sequential program statement is executed. The purpose of this subroutine is synchronization of program segments with the external time base. (Note that SB5 can be used, also, for general-purpose timing functions if the data acquisition channel is needed for other purposes. The user must simply connect the clock output to the FLAG input of the alternative I/O channel.)

EXPERIMENTAL

Computer Instrumentation

The digital computer system in this work was a Hewlett-Packard 2115A, equipped with 8K core memory, high speed paper tape input, ASR/33 Teletype, a 16-bit 20 usec DAC, and an interfaced Tektronix Model 601 oscilloscopic display. In addition, the computer has an interfaced data acquisition system and a general-purpose experimental interface capability. (These are described in detail below.) The complete computer system is mounted in a cabinet which has roll-around capability. The software used for laboratory on-line operation (PRTB) utilized the H.P. BASIC compiler, H.P. #201112A. This software is made available from the computer manufacturer. The program listings of the additions to the H.P. BASIC compiler made here to implement PRTB are available from the authors upon request.

Data Acquisition and General-Purpose Interface Hardware

A schematic diagram of the data acquisition system operated in conjunction with PRTB is shown in Figure 3. The analog-to-digital converter (ADC) used in the data acquisition system was manufactured by Digital Equipment Corporation (DEC), Maynard, Massachusetts, (#C-002, 10-bit, 35 μsec conversion time, 0 to -10.23 V. input range).

The programmable clock is constructed from a 10 MHz crystal controlled oscillator scaled down to useable frequency ranges with MSI (medium-scale integrated circuit) programmable countdown logic. A schematic diagram of this module is given in Figure 4, with a complete list of hardware components. Programmed clock pulses are available from 100 KHz to 0.002 Hz. Each can be divided by 2 or 5. The specific frequency is selected by a 5-bit output word which is decoded in the clock module by 2 Texas Instrument SN74151 digital multiplexers. In addition to the programmed output, each decade output from 1 MHz to 0.001 Hz is available externally by patchboard connection.

Also available on the data acquisition panel were an ENCODE output bit which could be set and cleared by the computer, and a FLAG input terminal to allow external setting of the FLAG bit on the computer Input/Output (I/O) channel used for data acquisition. The end-of-conversion (EOC) flip-flop of the ADC was normally connected to the FLAG input; however, it was possible to connect any appropriate externally generated signal to the FLAG input. Six bits of digital information could be transferred to or from the computer through the data acquisition panel using patchboard connections on the panel. Other generally useful functions available on the data acquisition panel included patchboard connection to various logical devices such as 16 AND, OR, and NOT gates, 4 flip-flops, 2 one-shots, 4 analog switches, 4 relay drivers, a Schmitt trigger, a track-and-hold amplifier, 6 indicator lamps, and 2 push button switches with Schmitt trigger outputs. (See Ref. 6 for a general discussion of characteristics of control logic modules.) DEC R-series logic (7) Flip-Chip cards and power supplies were used for logic functions. For interface design which required more sophistication or more logic elements than could be obtained on the data acquisition panel, a DEC patchboard R-series Logic Lab (7) was available. Most experiments did not require interface hardware beyond that available on the data acquisition panel. All input and output to the computer were buffered with level conversion devices on the H.P. 2115A system so that all external connections are compatible with DEC R-series positive logic (7). DEC W601 and W510 level conversion cards were used.

An interfaced digital I/O module was also available for general-purpose digital communication between computer and experiment. The characteristics of this module are illustrated in Figure 5. This digital I/O Module was connected to a different I/O channel than the data acquisition panel. Thus, a separate independent ENCODE output and FLAG input were available through this module. The use of the digital I/O capability has been described above.

RESULTS AND DISCUSSION

Evaluation of the PRTB System

For the purpose of illustrating the capabilities of the PRTB software for this publication, the laboratory computer systems and software were tested with two synthetically generated experimental outputs which simulated typical experimental data (4, 8). The first of these involved a transient experiment with a rapid exponential voltage decay output. The second illustration involved application of ensemble averaging to the analysis of an experimental output which provided a repetitive voltage waveform with superimposed large amplitude random noise.

The exponential decay waveform had a time constant the order of 0.1 sec. Data were taken at a constant rate for a fixed length of time, and no processing of the data was carried out. The data were acquired, stored, and the digitized data printed out at the completion of the experiment. The program used is given in Figure 6, and a typical output is given in Table I.

The waveform for the ensemble averaging experiment had a fundamental frequency the order of 10 Hz., with superimposed large amplitude random noise. The experiment was repeatable and could be triggered by computer output. The programming objectives included repetitive initiation of the experiment, synchronization of data acquisition with experimental output, and repetitive coherent summation of the digitized waveforms to accomplish ensemble averaging. Finally, when the experiment was completed the averaged data were normalized and plotted on the teletype terminal. The computer program used is given in Figure 7. Figure 8 shows the original output waveform and the results of ensemble averaging.

Observations

It can be seen from the above examples that actual implementation of the PRTB system is relatively straightforward. Moreover, for all its simplicity, considerable experimental measurement capability exists in the system.

The requirements for implementing the PRTB system in the undergraduate laboratory include the following:

(1) an introduction to the BASIC language for computational purposes. This requires 1 to 2 hours of lecture time. It also requires that students have access to the laboratory computer (or other computer facilities providing BASIC capability) during the first few weeks of the semester to handle homework and laboratory computations.

(2) an introduction to the fundamental concepts of on-line computer operation. This requires about 6 hours of lecture. These introduce the student to the whole field of on-line computer applications and the technological details with which they must be familiar to implement this approach in the laboratory. These details include an introduction to the fundamentals of digital logic, timing, and synchronization. This involves the discussion of simple gates (AND, OR, NOT), the flip-flop, the one-shot, and the analog switch. The functional characteristics of these devices, as well as of analog-to-digital and digital-to-analog converters and voltage amplifiers, for interface design are defined. No attempt is made to provide a rigorous understanding of the electronic principles or detailed circuitry for these devices. The emphasis is on the student being able to recognize and use the fundamental characteristics of these devices for interface design.

(3) finally, the actual laboratory assignments are carried out. These assignments are designed to introduce the student to the analytical technique and methodology, just as would be done in the absence of the laboratory computer instrumentation. In addition, the student is expected to utilize the on-line computer as a data acquisition and data processing tool.

The final question to be considered here is what specific benefits are derived by the student from the use of the laboratory computer in on-line experimentation. First of all, the student is obviously exposed to state-of-the-art technology in laboratory experimentation. Secondly, the student is encouraged to use more rigorous data processing approaches than previously feasible for laboratory assignments. Thirdly, the student becomes keenly aware of the factors which limit and define experimental accuracy and precision. Fourthly, the student is able to make a first hand comparison of the effectiveness of conventional and computerized laboratory methodology. And, finally, the student's interest in the science of quantitative chemical experimentation is strongly stimulated.

Future Work

Several different experiments involving on-line computer studies are currently being developed which are appropriate for incorporation into the undergraduate analytical laboratory. These include kinetic methods of analysis, amperometric titrations, coulometric titrations, potentiometric titrations, spectrophotometric analysis, fast-sweep polarographic analysis, and thermal analysis methods, in addition to synthetic and chromatographic experiments already developed (8, 9). Some of these experiments will be included in the introductory course and some in an advanced analytical course (senior level at Purdue).

It is also desired to further improve the PRTB software so that the experimental subroutine calls can be replaced by conversational macro-instructions.

ACKNOWLEDGEMENTS

This work was supported by a grant from the National Science Foundation Office of Computing Activities, Grant No. GJ-428. The authors wish to thank Dr. David O. Jones for technical advice, and Professors L. B. Rogers, H. L. Pardue, and F. E. Lytle for their respective contributions to the suggestion, encouragement, criticism, and technical support of this program.

REFERENCES

1. J. W. Frazer, Anal. Chem., 40, 26A (1968).
2. N. R. Kuzel, H. E. Roudebush, C. E. Stevenson, J. Pharmaceut. Sci., 58, 381 (1969).
3. S. P. Perone, J. Chrom. Sci., 7, 714 (1969).
4. S. P. Perone, J. Chem. Ed., 47, 105 (1970).
5. J. G. Kemeny and T. E. Kurtz, "BASIC Programming," John Wiley & Sons, N. Y., 1967.
6. H. V. Malmstadt and C. G. Enke, "Digital Electronics for Scientists," W. A. Benjamin, Inc., N. Y., 1969.
7. "Digital Logic Handbook," Digital Equipment Corporation, Maynard, Massachusetts, 1968.
8. D. O. Jones, to be submitted for publication.
9. S. P. Perone and J. F. Eagleston, to be submitted for publication.

FOOTNOTES

* Trade Mark, Digital Equipment Corporation, Maynard, Massachusetts Massachusetts.

LIST OF FIGURES

Figure 1. Schematic Diagram of On-Line Computer Configuration.

Figure 2. Synchronization of Clock Output with Computer and Experimental Events.

A. Direct computer control of Clock Enable input.
B. Clock Enable controlled by AND gate output. Clock enabled when both computer command and experimental start are seen at AND gate inputs. The "RUN" flip-flop output follows the status of the experiment; it is set "True" when the experiment starts and "false" when the experiment ends. Thus, the Clock is disabled when the experiment ends.

Figure 3. Typical PRTB Experimental Data Acquisition Set-up.

Previously undefined symbols include:

I/O Bus - Input/Output hardware of the computer, one channel of which is used for data acquisition system.
16-bit duplex register - A Hewlett-Packard #02116-6195 16-bit duplex interface buffer register card, also providing output ENCODE command and input FLAG signal.

Figure 4. Schematic Diagram for PRTB Programmable Clock.

Components:

Device No.	Manufacturer	Description
F910959	Fairchild	Hex inverter-six Level Converters, High level to +4 to +20 V.
MC7475P	Motorola	Quad latch--4 bistable latches
SN74151	Texas Instrument	Data Selector/ Multiplexer
F930059	Fairchild	4-bit shift register
F900359	Fairchild	Triple 3-input Nand Gate

Figure 5. Digital I/O Interface Module for PRTB.

Figure 6. Program for Data Acquisition from Exponential Decay Function.

Figure 7. Program for Ensemble Averaging Experiment.

Figure 8. Data from Ensemble Averaging Experiment, plotted on Teletype.

A. Original data (single cycle).
B. Data after 100 averaging cycles.

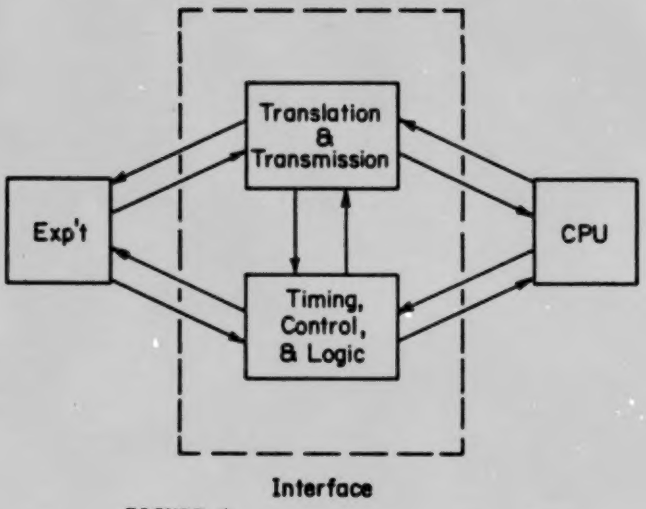

Translation & Transmission
Exp't
CPU
Timing, Control, & Logic
Interface

FIGURE 1

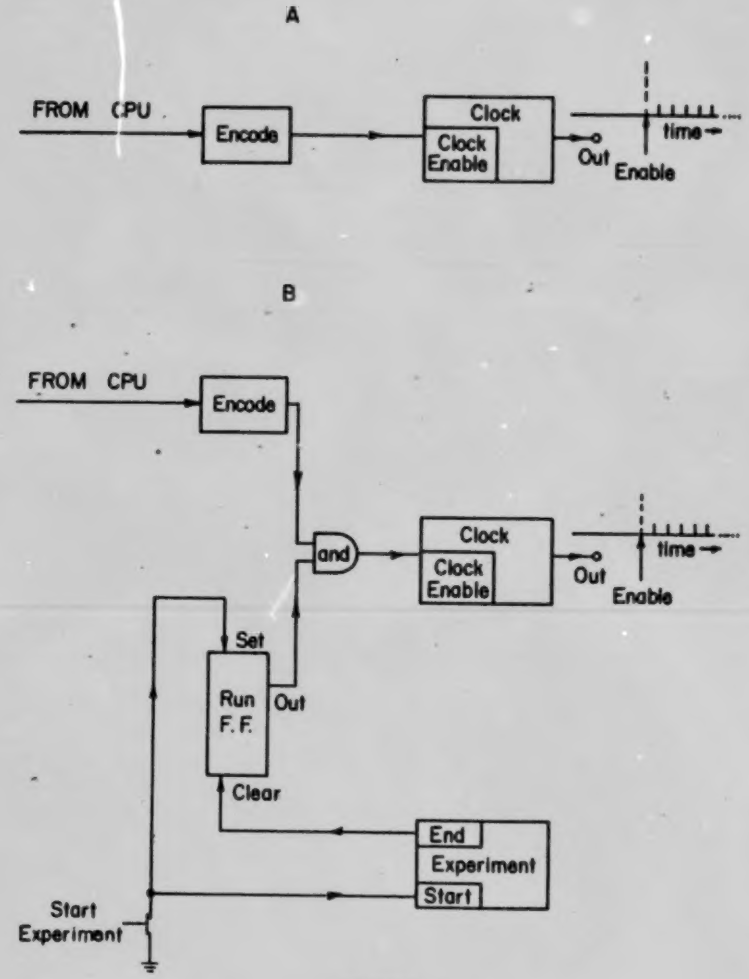

A
FROM CPU
Encode
Clock
Clock Enable
Out
time
Enable
B
FROM CPU
Encode
and
Clock
Clock Enable
Out
time
Enable
Set
Run F.F.
Out
Clear
End
Experiment
Start
Start Experiment

FIGURE 2

FIGURE 3

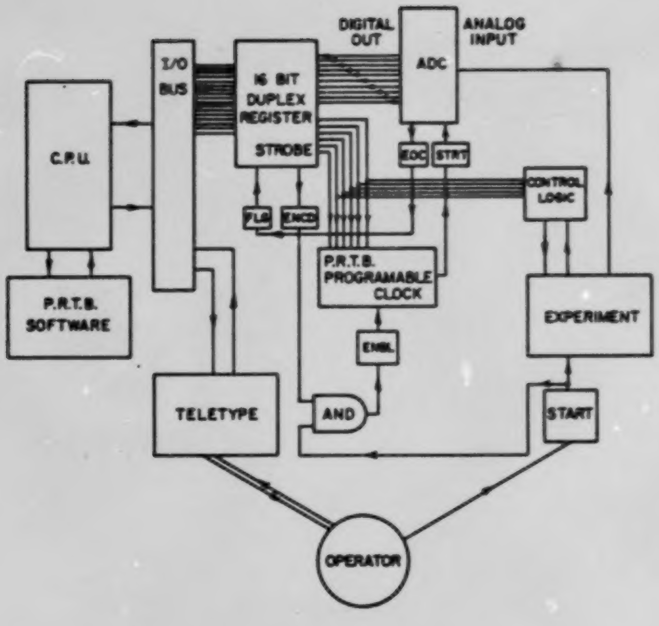

DIGITAL OUT
ANALOG INPUT
ADC
I/O BUS
16 BIT DUPLEX REGISTER
STROBE
C.P.U.
EOC
STRT
CONTROL LOGIC
FLG
ENCD
P.R.T.B. PROGRAMABLE CLOCK
P.R.T.B. SOFTWARE
EXPERIMENT
ENBL
TELETYPE
AND
START
OPERATOR

FIGURE 4

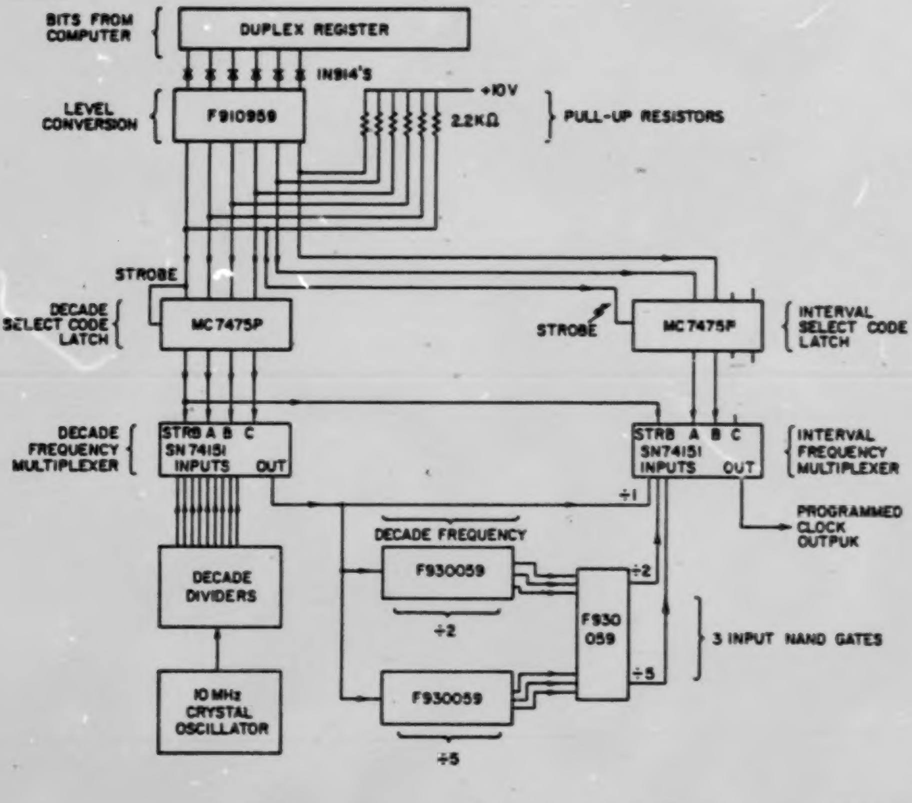

BITS FROM COMPUTER
DUPLEX REGISTER
IN914'S
LEVEL CONVERSION
F910959
+10V
2.2KΩ
PULL-UP RESISTORS
STROBE
DECADE SELECT CODE LATCH
MC7475P
STROBE
MC7475P
INTERVAL SELECT CODE LATCH
DECADE FREQUENCY MULTIPLEXER
STRB A B C
SN74151
INPUTS OUT
STRB A B C
SN74151
INPUTS OUT
INTERVAL FREQUENCY MULTIPLEXER
÷1
PROGRAMMED CLOCK OUTPUK
DECADE FREQUENCY
F930059
÷2
÷2
F930 059
3 INPUT NAND GATES
÷5
F930059
÷5
DECADE DIVIDERS
10 MHz CRYSTAL OSCILLATOR

FIGURE 5

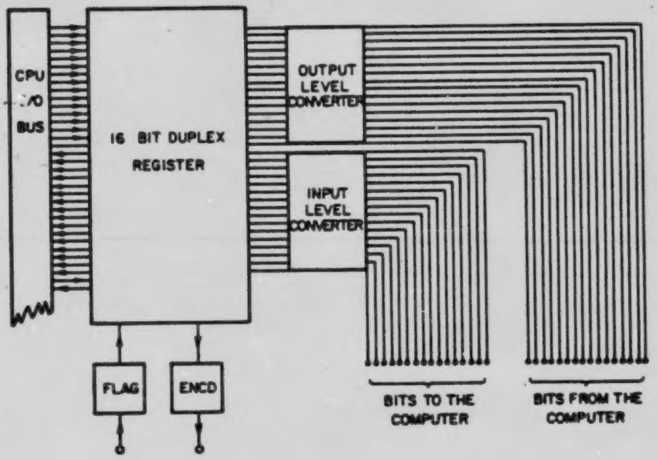

CPU
I/O
BUS
16 BIT DUPLEX
REGISTER
OUTPUT
LEVEL
CONVERTER
INPUT
LEVEL
CONVERTER
FLAG
ENCD
BITS TO THE
COMPUTER
BITS FROM THE
COMPUTER

FIGURE 6

```
READY
LIST
10 REM THIS IS A PROGRAM THAT WILL PRINT OUT THE DECIMAL NUMBER
20 REM EQUIVALENT OF AN ANALOG EXPONENTIAL DECAY CURVE.
30 DIM X(250)
40 LET F=100
50 LET T=250
60 CALL (1,X(1),F)
70 CALL (7,T)
80 FOR I=1  TO  250
90 PRINT X(I);
100 NEXT I
110 END
```

FIGURE 7

```
READY
LIST
1 REM THIS PROGRAM IS AN ENSEMBLE AVERAGING ROUTINE--FOLLOW
2 REM THE DIRECTIONS PRINTED OUT AFTER PRESSING RUN.THE
3 REM FREQUENCY ORIGINALLY SPECIFIED IS 1000 HZ.
10 DIM X(100),Y(100)
20 LET F=1000
30 PRINT "THE # OF PTS EACH RUN =";
31 INPUT T
40 PRINT THE # OF RUNS =";
41 INPUT C
50 MAT Y=ZER
60 FOR I=1 TO C
70 CALL(1,X(1),F)
80 CALL(7,T)
90 FOR J=1 TO T
100 LET Y(J)=Y(J)+X(J)
110 NEXT J
120 NEXT I
200 FOR K=1 TO T
210 LET E=Y(K)/(C*10)
220 IF E>72 THEN 300
230 PRINT "*";TAB(E);"*"
240 NEXT K
250 STOP
300 LET E=71
310 GOTO 230
400 END
```

3 1

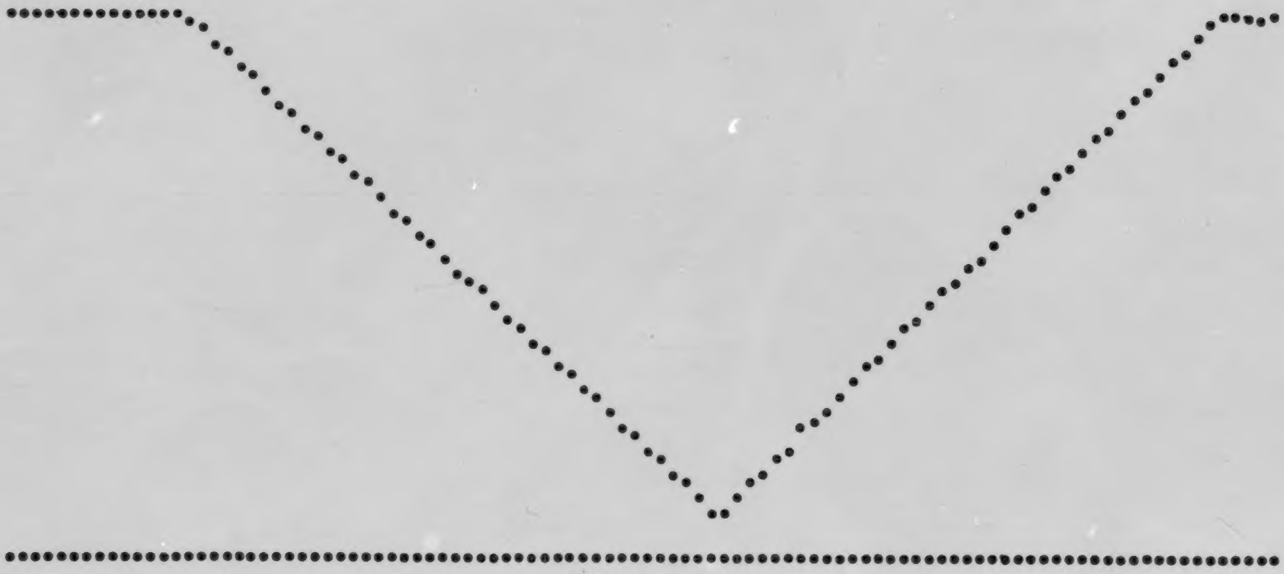


FIGURE 8 (a)

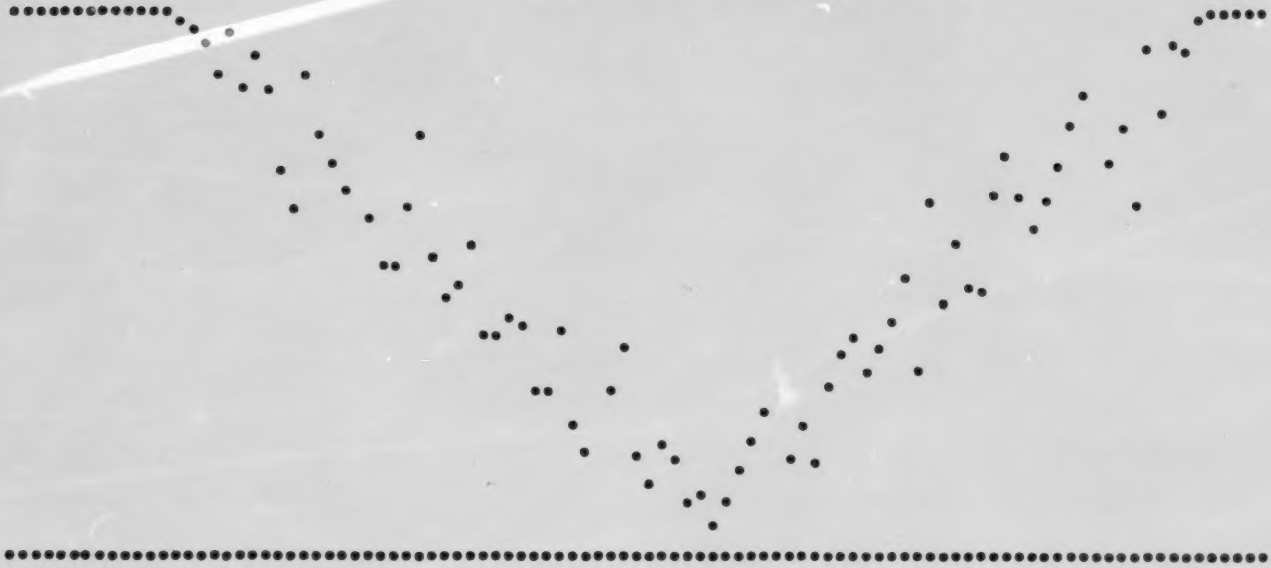


FIGURE 8 (b)

COMPUTERIZED EXPERIMENTS USING AN A/D CONVERTER

John H. Karl
Wisconsin State University
Oshkosh, Wisconsin

The Physics Department at WSU-O has placed emphasis on the use of the computer as a computer in its introduction into the physics curriculum. Due to the large amount of resources required to embark upon an experimental education project in computer assisted instruction (CAI) or its related areas, we have decided to confine our initial efforts to instructing both our non-physics majors and our majors in the use of this important powerful tool.

Our efforts can be divided into two areas: (1) stand-alone processing of numerical data and (2) on-line data collecting and data processing. We feel that since our physics majors go on to advanced research projects in either graduate schools or industrial laboratories, they should have as much grounding in state-of-the-art experimental techniques as possible.

Our purpose here is to report on the introduction of on-line data collection and data processing techniques into our intermediate physics laboratory.

Using a minimum configuration PDP-8L (4K) and a digital Equipment AD01 analog to digital converter we have developed an interfacing to two existing physics experiments in our intermediate laboratory. The first is a microwave apparatus used to simulate Bragg diffraction of X-rays in crystals. The second experiment is a nuclear magnetic resonance (NMR) station. Planned, but not yet developed, experiments include Fourier analysis using the fast Fourier transform, pulse height analysis, and the study of the Poisson distribution in nuclear decay. Data collection on the Bragg diffraction apparatus, signal averaging on the NMR station, and digital filtering of the NMR signal have been developed and will be discussed here.

FACILITIES

All of the above experiments are based on using the AD01 A/D converter in conjunction with the PDP-8/L computer. The 10 bit A/D converter has up to 32 analog channels selected by program control. The conversion thruput time is 16 usec with a 10 usec aperture. The accuracy is ± .05% full scale over a 10 volt range which can be divided into four smaller ranges by program control.

Programs were written in PAL-III assembly language so that the A/D converter could be called using the FNEW function in FOCAL, a DEC (Digital Equipment) supplied conversational interpretive language for the PDP-8. All of the students in intermediate laboratory have been exposed to FORTRAN programming to some extent so that using FOCAL requires little effort. Thus time can be spent on the techniques of on-line data collection and processing rather than on learning a new assembly language. An additional bonus is the interactive character of the language which is especially valuable in on-line operation. The price paid for using FOCAL is the large amount of core it requires and the large amount of execution time involved. There is a maximum of 100 locations left for the users program. Calling the A/D converter through a FOCAL subroutine which selects scale range, channel, and does the conversion to the FOCAL decminal floating point word format results in a thruput time of 15 msec compared to the basic 16 usec thruput of the AD01 alone. This relatively slow thruput rate can be tolerated in many research and instructional experiments.

The Bragg Microwave-Diffraction Experiment

The Bragg microwave diffraction apparatus used is commercially available from Welch Scientific Company. This apparatus was chosen for our first experiment with on-line data collection and processing because of its simplicity and because of the tedium it presented in the manual collection of data. The apparatus, shown in Figure 1, consists of a 4 x 4 x 4 array of steel spheres imbedded in a plastic foam matrix, a three cm modulated (1000 cycle) microwave klystron transmitter, and a crystal diode receiver connected to a tuned AC voltmeter (HP 415B). The recorder output of the AC voltmeter is interfaced (biased with an inversion of polarity) to the AD01. A slide wire variable resistor attached to the degree circle at the base of the apparatus allows voltage pickoffs by contacts connected to the transmitter, receiver, and "crystal" mount.

A FOCAL program (see Figure 2) continuously reads the output of the detector, the angle of the transmitter, receiver, and crystal. If the relative orientation of the transmitter, receiver, and crystal satisfies the Bragg condition, the detector output is stored in an array vs. the Bragg angle. The student moves the arms that support the transmitter and receiver so as to keep them at equal angles about a fixed reference point. When the computer has collected all the acceptable data the program types out "Data Complete" and plots the output vs. angle. The results of such a plot are shown in Figure 3. The student has the responsibility of determining the angle at which the Bragg maximum occurs and calculating the distance between reflecting planes from the Bragg Law. This of course can be accomplished by a FOCAL program written by him. Due to the high data collection rate the student can extend the experiment in several ways2 For example, by rotating the entire transmitter, receiver, and crystal combination a fixed amount, spurious maxima from room reflections can be distinguished from the Bragg diffraction. The Bragg maxima will occur at the same angles, but the background spurious responses will be shifted to new positions. Alternately data collected with the "crystal" removed could be subtracted from the "crystal"-plus-background data. Such ideas will be left for the student to explore next semester (Spring, 1971) when the "computerized" version of this experiment will be introduced into our intermediate laboratory.

The NMR Experiment

The second experiment used for on-line data collection and processing in Atomic Laboratories Inc.'s combination nuclear magnetic resonance (NMR) and electron spin resonance (ESR) spectrometer. (Cenco No. 71913). A current regulated supply feeding an Atomic Laboratories' 3" magnet provides the DC magnetic field. Since the NMR experiment has rather poor signal to noise ratio, it was selected for introducing the student to the technique of signal averaging. Data previously used in this experiment was displayed on an oscilloscope screen. (See Figure 4.)

Since the magnetic field in this apparatus is swept by a 60 cycle line source, the coherence required for signal averaging can be obtained by using line voltage as a timing reference. Advantage was taken of the fact that the output of the unipolar AD01 is zero when the input signal is negative. Thus the 60 cycle AC line voltage looks like a half rectified wave when viewed through the A/D converter.

A second PAL-III program was written, (copies are available from the author), that can be called from FOCAL to provide the required timing. Arguments of the subroutine are the scale range, the channel number, and the time at which the channel is to be sampled. The time parameter is the number of 400 usec intervals elapsed after the 60 cycle line frequency passes through zero with positive slope. The period of a 60 cycle wave form can then be divided into about 40 time slots. The timing is determined by searching for the first positive value of the rectified AC line after the line was at zero. Then the program enters a no operation 400 usec loop for the desired number of times and then samples the signal channel called for by FOCAL. Consistency of timing obtained in this manner is ± 14 usec with a constant time delay from the 60 cycle zero to the first time bin of 27 usec.

A FOCAL program calls the A/D converter timing routine through the function named FNEW (Figure 5). This FOCAL program fills 46 time bins with N values to average out the noise. The resulting array of signal vs. time is scaled and plotted on the teletype. Results for N = 1 and N = 500 are shown in Figures 6 and 7. Since the aperture of the A/D converter is 16 usec and the data is taken at rather widely spaced intervals (every 400 usec), the results for N = 1 appear poorer than the scope display. The results for N = 500 are rather impressive when compared to the scope display. Considerably more dramatic results could be presented if a plotter output were available. The signal to noise ratio of the proton resonance in tap water is about one-to-one in our NMR rig. It is hoped that the students will be able to enhance this signal considerably by using this signal averaging technique next semester in intermediate laboratory2 Aside from the usual NMR studies, the students will be expected to study the dependence of signal to noise ratio on the number of averages. Another problem the student faces is that of plotting the resonance vs. magnetic field, as is customary, rather than vs. time as is done here. This can readily be accomplished within the PDP-8 FOCAL language.

Digital Filtering

From the NMR data a considerable 60 cycle component can be seen (perhaps most easily in the oscilloscope trace, Figure 4). Of the various techniques which could be used to eliminate this component from the signal, possibly the least known to our students and physics students in general, is digital filtering. With the greatly increased number of on-line applications of digital computers for collecting and processing data in the fields of science and engineering, digital filtering has become increasingly important. The NMR signal provides an opportunity for introducing digital filtering techniques into our physics curriculum.

The object is to remove the 60 cycle noise and other low frequency components that may be present. Hence a high pass filter is called for with a low frequency cut-off of, say, 120 cycles. One way this can be accomplished in the computer is by calculating the Fourier components of the averaged signal by numerical integration. Then the filtered function can be synthesized by summing the components and omitting the 60 cycle (the fundamental) and the 120 cycle component. Using this approach, one must truncate the Fourier series somewhere and therefore one is led to a band-pass filter with some high frequency cut-off. From inspection of the NMR data, it can be seen that the resonance width is about one-eighth of the 60 cycle period or about 2 msec or 5 Kc which requires about 85 harmonics. This can readily be accomplished within the limits of the FOCAL language.

We are currently writing digital filtering programs of this type for experimentation within our intermediate laboratory. Many facets of digital filter design may be explored by the students experimentally. For example, the limitations of the numerical integration will result in less than ideal response. The student can determine the actual response of the filter by experiment. The effect of various cut-off frequencies on the signal may be studied in regard to both low and high frequency noise. This may also be an excellent place to introduce the student to the fast Fourier transform algorithm. Introductory material and standard reference works on digital filtering will be provided for the student.

CONCLUSION

The above "computerized" experiments will be "student tested" for the first time this spring semester, 1971. We believe that we have made an important step toward introducing modern data treatment techniques into the physics curriculum. It is such technical skills that allow the physicist to bridge disciplines and make himself valuable to a wide spectrum of employers.

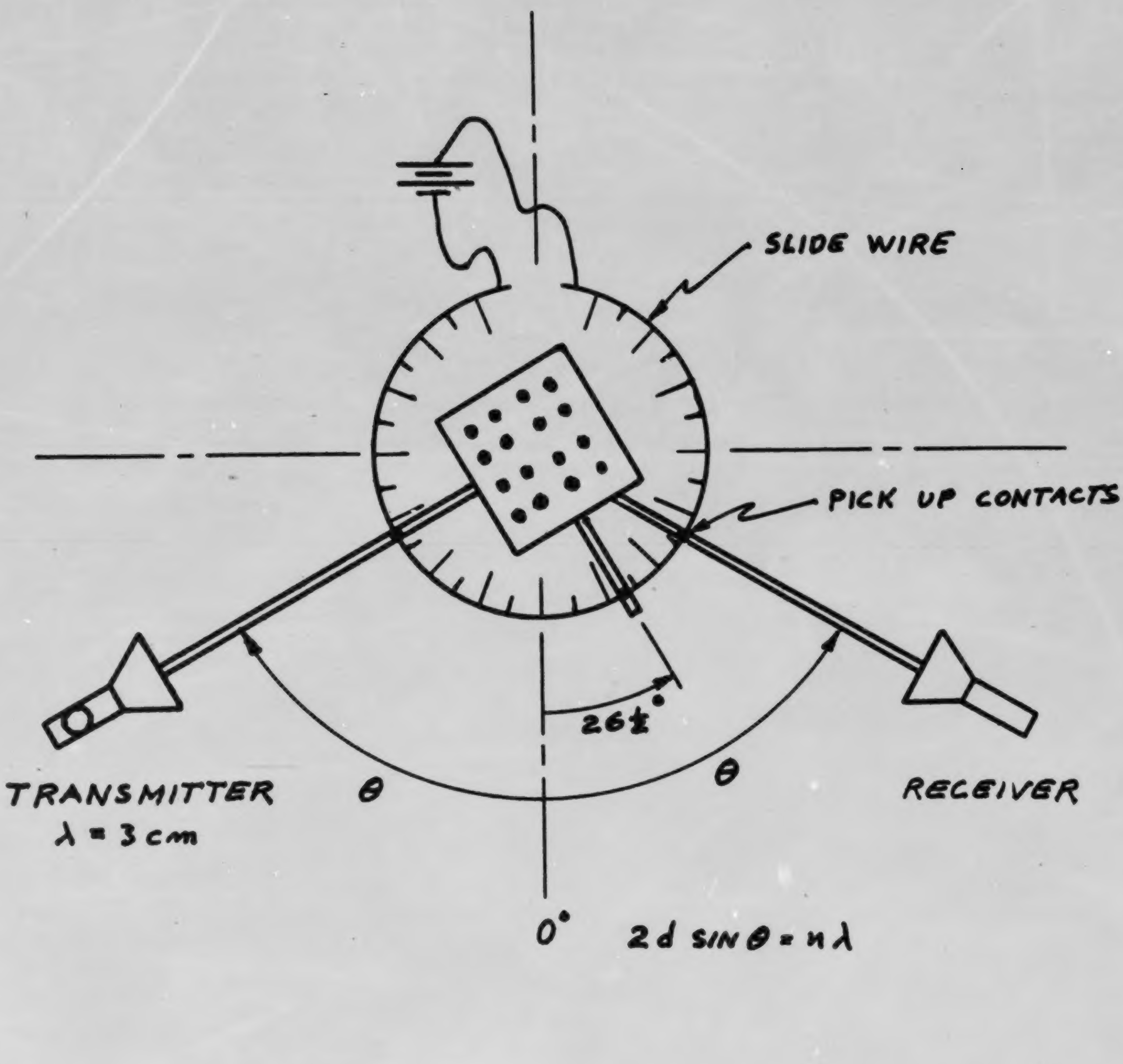

SLIDE WIRE
PICK UP CONTACTS
26½°
TRANSMITTER
λ = 3 cm
θ
θ
RECEIVER
0°
2d SIN θ = n λ

Fig. 1

Bragg diffraction apparatus. Crystal is shown offset 26 1/2° for the (210) lattice orientation.

```
C-FØCAL, 1969

01.10 T "PRØGRAM FØR BRAGG DIFFRACTIØN"!!!!

02.05 S R1=1; S R2=1
02.10 A "MAX DEGREES",UD;A "MILLER ØFFSET",MØ;A "PRECISIØN",PP
02.15 T "DIGITIZE MAXA"; A DUMMY
02.20 S DV=2*UD/(FNEW(R1,2)-FNEW(R1,1));T "DIGITIZE MINA"; A DUMMY
02.25 S LD=FITR((FNEW(R1,2)-FNEW(R1,1))*DV)
02.30 F I=LD,PP,UD;S D(I)=0.0

03.10 S X=FNEW(R2,0); S T=FITR(FNEW(R1,1)*DV)
03.11 S R=FITR(FNEW(R1,2))*DV;S C=FITR(FNEW(R1,3)*DV)
03.15 I (FABS(C-(T+R)/2-MØ)-PP) 3.20,3.20,3.10
03.20 S A=(R-T)/2;I (A-UD) 3.30,3.30,3.10
03.30 I (A-LD) 3.10,3.40,3.40
03.40 S D(A)=X;F I=LD,PP,UD;D 4.0

04.10 I (D(I)) 5.60,3.10,4.20
04.20 C

05.10 T "DATA CØMPLETE";T !!!!!
05.50 GØTØ 10.10
05.60 T "ERRØR ØCCURED";QUIT

10.10 F I=LD,UD;DØ 11.0

11.10 T %3,I;T %7.07,D(I);F J=0,D(I);T" ";T "*"!

12.10 T!!!!!
```

Fig. 2

FOCAL program for the Bragg diffraction experiment. The function FNEW (A,B) calls the A/D converter for range A and channel B.

```
20 0.1678      *
21 0.1476     *
22 0.1427     *
23 0.1738      *
24 0.1543     *
25 0.1444     *
26 0.1405     *
27 0.1553     *
28 0.1430     *
29 0.1397     *
30 0.1721      *
31 0.1792      *
32 0.1899      *
33 0.2168       *
34 0.2096       *
35 0.1788      *
36 0.1367     *
37 0.1275     *
38 0.1343     *
39 0.1671      *
40 0.2356       *
41 0.4294           *
42 0.5579              *
43 0.8382                      *
44 0.8536                       *
45 0.9819                          *
46 0.7172                   *
47 0.4722            *
48 0.3125         *
49 0.1729      *
50 0.1541     *
51 0.1648      *
52 0.1992      *
53 0.2254       *
54 0.2502        *
55 0.1732      *
56 0.1305     *
57 0.1319     *
58 0.1573     *
59 0.1986      *
60 0.2377       *
61 0.2295       *
62 0.1432     *
63 0.1387     *
64 0.1623      *
65 0.4251           *
66 0.6140               *
67 0.9868                          *
68 1.6304                                            *
```

Fig. 3

Teletype plot of Bragg diffraction data. The numbers in the first column are the angle θ of fig. 1 in degrees. The second column is the detector output in volts. This plot is for the (100) orientation. Note the secondary maxima on each side of the primary peak.

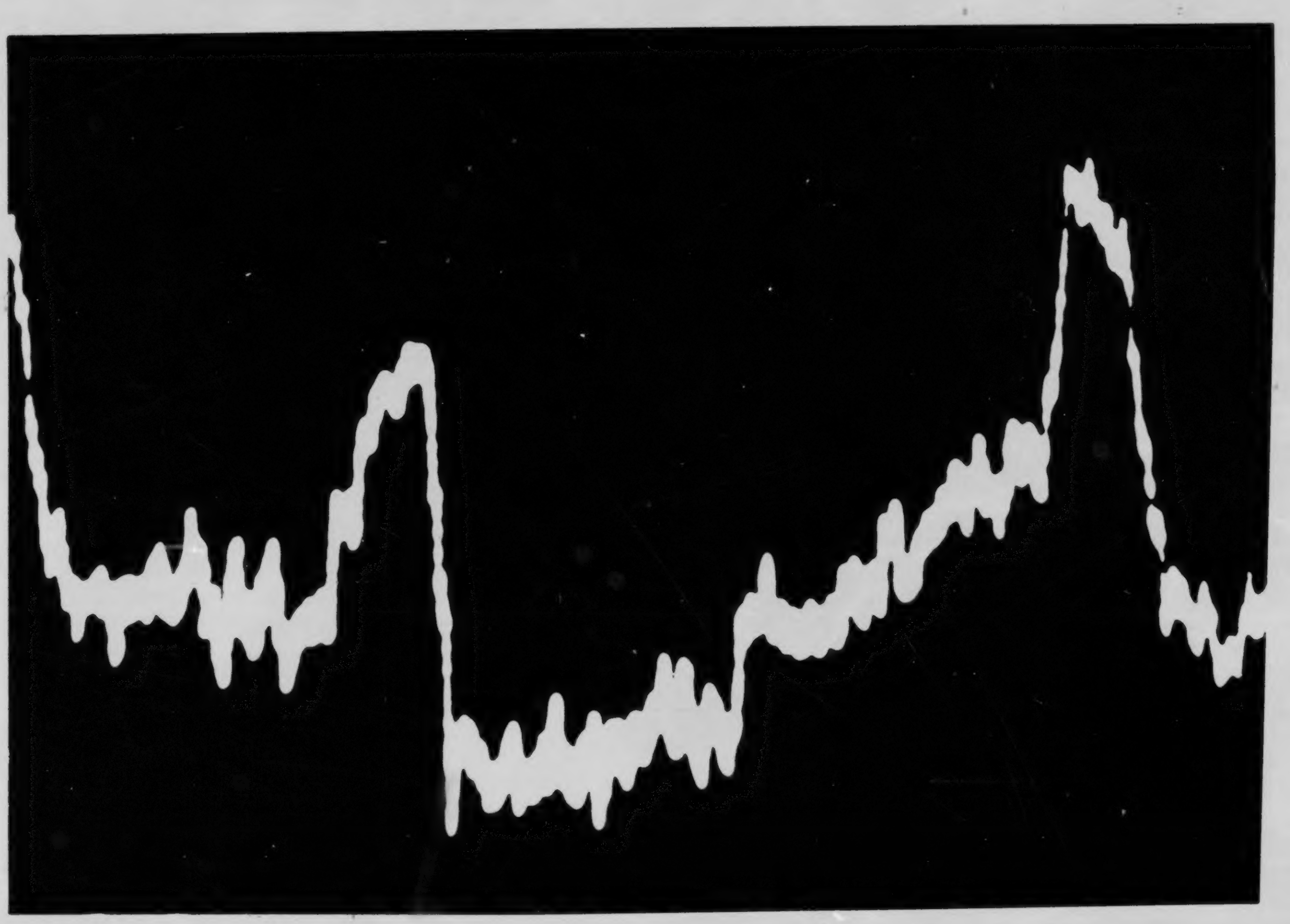


Fig. 4

Oscilloscope trace of the proton resonance in a copper sulfate solution. The trace covers about 16 uses. Note the 60 cycle background and the high frequency noise.

```
01.05 A " N",N," I",BI," F",BF
01.06 T !,!,!,!,!,!,!,!,!,!,!,!
01.07 T " NUMBER ØF EVENTS",N,!
01.08 T " INITIAL BIN",BI,!," FINAL BIN",BF,!
01.20 F J=1,N; D 2
01.25 F I=BI,BF; D 4
01.30 G 5.01

02.05 F I=BI,BF; D 3

03.05 S A(I)=A(I)+FNEW(0,6,I)

04.05 S A(I)=A(I)/N

05.01 D 7; D 9; S G=50/(UA-LA)
05.05 F I=BI,BF; D 6
05.10 G 1.06

06.05 T %3,I,%8.4,A(I); F K=0,G*(A(I)-LA); T " "
06.10 T "*",!

07.50 S UA=0; F I=BI,BF; D 8

08.10 I (A(I)-UA) 8.3
08.20 S UA=A(I)
08.30 C

09.50 S LA=UA; F I=BI,BF; D 10

10.10 I (LA-A(I)) 10.3
10.20 S LA=A(I)
10.30 C

*
```

Fig. 5

The FOCAL time averaging program. The function FNEW in line 3.05 calls the A/D converter. Arguments are scale range, channel number, and the time of sampling.

```
NUMBER 0F EVENTS 1.0000000
INITIAL BIN 0.0000000
FINAL BIN 45.000000
  0 1.3281300       *
  1 1.7890600            *
  2 1.9921900              *
  3 1.0546900     *
  4 1.9218800             *
  5 2.4531300                 *
  6 1.9765600            *
  7 1.6015600         *
  8 1.8203100           *
  9 2.3125000               *
 10 1.4921900        *
 11 2.0625000             *
 12 2.3281300               *
 13 2.0546900             *
 14 2.5234400                 *
 15 5.0625000                                     *
 16 2.5546900                 *
 17 4.1406300                            *
 18 4.5312500                               *
 19 5.2812500                                      *
 20 2.0859400             *
 21 1.8593800          *
 22 1.7187500         *
 23 0.7578130 *
 24 2.1640600              *
 25 2.6406300                  *
 26 1.7265600         *
 27 1.5937500        *
 28 1.6093800        *
 29 1.3359400      *
 30 1.5156300        *
 31 1.9375000           *
 32 2.6093800                  *
 33 3.6484400                           *
 34 3.1796900                       *
 35 2.1171900             *
 36 2.6406300                  *
 37 3.4531300                          *
 38 6.0546900                                              *
 39 6.8828100                                                    *
 40 5.7578100                                          *
 41 4.0312500                              *
 42 4.9609400                                     *
 43 3.0312500                     *
 44 1.8515600        *
 45 1.8593800        *
```

Fig. 6

The NMR signal with one data sample in each bin. The 46 bins cover approximately one 60 cycle period. The first column is the bin number; the second is the detector output in volts.

```
NUMBER ØF EVENTS 500.00000
INITIAL BIN 0.0000000
FINAL BIN 45.000000
  0 0.0331094  *
  1 0.0498594  *
  2 0.0719062   *
  3 0.0990312    *
  4 0.1041560    *
  5 0.1154530     *
  6 0.1380470     *
  7 0.1533280      *
  8 0.1997970        *
  9 0.3154840             *
 10 0.5306090                     *
 11 0.9141410                                   *
 12 1.2342700                                              *
 13 1.3621700                                                  *
 14 1.3627000                                                  *
 15 1.0471900                                       *
 16 0.6382030                          *
 17 0.3415470              *
 18 0.1118280     *
 19 0.0462344  *
 20 0.0188281 *
 21 0.0175625 *
 22 0.0282656 *
 23 0.0353906  *
 24 0.0498438  *
 25 0.0739844   *
 26 0.0900469    *
 27 0.1767660       *
 28 0.3458440              *
 29 0.6738280                           *
 30 0.9344370                                    *
 31 1.1682700                                           *
 32 1.2415600                                              *
 33 1.0659100                                        *
 34 0.5890160                      *
 35 0.2551560          *
 36 0.0111719 *
 37 0.0021406 *
 38 0.0039688 *
 39 0.0048906 *
 40 0.0066719 *
 41 0.0046719 *
 42 0.0048594 *
 43 0.0121250 *
 44 0.0283437 *
 45 0.0436406  *
```

Fig. 7

The time averaged NMR signal with 500 samples in each bin. The scale is such that bins higher than 35 are plotted as zero on the Teletype. Fig's 6 and 7 are phase shifted from fig. 4.

SOME STUDENT LABORATORY EXPERIMENTS IN DATA SAMPLING BY MINICOMPUTER

Dr. Samuel A. Elder
United States Naval Academy
Annapolis, Maryland 21402

I. INTRODUCTION

Several years ago the one-semester acoustics course taught by the Physics Department at the United States Naval Academy was restructured to make it computer-oriented. At that time the course met for three lectures a week, with no laboratory. Computer involvement was via homework assignments on time-shared teletype terminals. This effort was reported on last year at the first Conference on Computers in the Undergraduate Curricula at Iowa City. (1)

During the Fall of 1970 acoustics became a four-credit-hour course with the addition of a weekly two-hour laboratory. The Naval Academy is well equipped for lab work in this area because of the strong faculty research interest in Acoustics. The Department has its own anechoic chamber facility for acoustic calibrations. In addition there is a Digital Analyzer system, based on the PDP8/I minicomputer. The presence of this latter equipment offered the possibility of student exercises in real time digital data sampling. Heretofore data bases for assignments on Fourier Analysis, autocorrelation, etc., had been rather artificially contrived, since it is not possible to feed a live acoustic signal into the teletype!

A strong reason for the introduction of hardware-centered assignments was the feeling that something valuable is always gained from student "hands on" experience. The time-shared teletype system comes very close to this ideal. In a smoothly running system time-sharing is little different from direct one-man operation from the point of view of the operator. However a student may be a good programmer and have almost no understanding of computer hardware. The teletype always talks to him in a high-level language similar to his own and never mentions sw6tch registers, binary arithmetic or Schmitt triggers. I won't argue that this isn't a good thing for student programmers. But when a computer is to be used as a data acquisition device in real time it becomes a part of the laboratory measurement apparatus. As in the case of voltmeters, amplifiers, and other electronic aids, its effect on the measurement needs to be completely understood. This can only be accomplished by examining the precise way in which it quantizes a continuous signal into a time series of binary integers, and then further digests it. Therefore we decided to develop a student laboratory experiment specifically designed to examine the process of computer data sampling, and also its inverse, computer function generation.

II. HARDWARE

1. Overall Configuration.

The available hardware consisted of a PDP8/I minicomputer equipped with ASR33 Teletype, a high speed analog-to-digital converter, two channels of digital-to-analog output, and a dual channel low-pass analog filter. Additional circuitry was added to the "front end", using D.E.C. logic modules, to permit external clock synchronization of the sampling rate. The minimum configuration represents an investment, in current market value, of about $11,000. The computer console shown in Figure 1 has some additional features which make it convenient to use, though they are not essential to the present operation. Visible on the instrument rack, for example, is a high speed paper tape reader. Inside the chassis there are a homemade input multiplexer (permitting dual-channel data taking), a hardware multiply option, and an extra 4096-word core memory storage unit.

2. Central Processer Console.

The central processor contains the core memory, registers, and control unit. The standard PDP/8 has a basic random access memory made up of 4096 twelve-bit binary words. Direct communication with the processor can be made through a twelve-bit switch register on the console panel. The switches are grouped into four color coded sections of three bits each, facilitating easy translation between binary and octal-coded register words. (See Figure 2). All arithmetic is performed in the accumulator register, one of four "light indicator" registers on the front panel. Any integer from 0 to 4095_{10} can be manually entered into the computer memory by programming the computer to read its own switch register into the accumulator. The number 4095 happens to be the largest integer that can be represented in a 12-bit binary word (i.e. $2^{12} = 4095+1$). This sets a limit on the dynamic range of numbers that can be stored as single memory-words. The "speed" of a computer is determined by execution times of machine language commands. For the PDP8/I the shortest complete command takes about 1.5 microseconds. This makes it a relatively fast computer, and it is possible to do some simultaneous processing of incoming data even when sampling is proceeding at a rather high rate. Mini's are not at all inferior to larger computers in this respect.

3. Analog-to-Digital Converter.

The analog-to-digital or "A/D" converter used in our configuration (D.E.C. Model AD08A) is able to sample voltage signals in the range -5 to +5 volts at rates up to 100,000 per second. At the input of the converter is a "sample-and-hold" amplifier (D.E.C. Model A400), which is a device for improving the accuracy of sampled signals. It requires about 10 microseconds to perform a single conversion. The sample-and-hold amplifier samples the analog voltage very quickly and "holds" it for about 10 microseconds so that the conversion process is not affected by any changes in the incoming signal (See Figure 3b). Digitized numbers are represented in 10-bit words and therefore have a possible dynamic range of 1 to 1023 (i.e., $2^{10} = 1023+1$). Thus the best voltage resolution is about 10 mv. Incoming numbers are temporarily stored in the accumulator. Inputs from the A/D converter are left-justified before storage. Since the computer deals in 12-bit words this means that a stored number begins at the left-most bit and the last two bits are ignored. The actual resolution after the conversion process is limited by the amount that the signal was able to change over the short time of active sampling, or "aperture" time. For our model the aperture time is 150 nanoseconds. This permits full 10-bit accuracy to be maintained up to a signal frequency of about 2500 Hz. For higher frequency signals the resolution of the digitized signal decreases with frequency.

4. Digital-to-Analog Converter.

The digital-to-analog or D/A converter gives the computer a real time output channel which can be displayed on an oscilloscope screen. Our configuration (D.E.C. Model VC8/I) has two 10-bit channels either of which can convert a number held in the accumulator to a voltage signal. Fullscale output is 2 volts. Before a number in the accumulator can be converted it must be right-justified. That is, only the last 10 bits of each word are used for output. The converted voltage is maintained at a fixed level at the output of the converter until the next sample is received. When read out at the sampling rate, therefore, the converted signal maintains the discrete character of the stored time series, only in analog form (See Figure 3c).

5. Low-Pass Analog Filter.

As will be discussed later, signals which contain energy at frequencies more than one-half the sampling rate suffer "aliasing" distortion in the sampling process. This is due to a folding back of the upper part of the spectrum into the lower producing spurious or "alias" results. To avoid such effects one must either sample at a very high rate or prefilter the signal before sampling. Since high sampling rates create a memory storage problem, filtering is usually preferable. Our system utilizes a dual variable low-pass filter (Reaction Instruments Inc.) each channel of which can be set at any frequency from .001 to 99,900 Hz. The filter response is of the Chebyshev type, which is noted for sharp cut-off. The second filter channel is used at the D/A output to remove sampling-rate noise.

6. Teletype.

An ASR 33 teletype console is included in the standard PDP/8 package. It is, of course, the main interface between the operator and computer in editing and running programs. In addition, however, it may serve as a sense switch manual in controlling real time data input, a selection manual for function generation, and a printout device for processed data. When a decimal digit is entered from the teletype keyboard, it appears in the accumulator, right-justified, as an ASCII-coded binary number. This can be easily reduced to standard binary form by subtracting the ASCII equivalent of zero. By an inverse process a number in memory can be prepared for exit to the teletype printer. The sense switch capability derives from the keyboard skip-if-flag instruction KSF, which is able to detect when a key is struck.

III. STUDENT EXPERIMENTS

1. Learning to Manipulate Peripherals.

All student programming was done either in machine language or in symbolic assembler code. Although there are several high level language compilers available in the PDP/8 software package (FORTRAN, FOCAL, and in some cases BASIC) programs which involve primarily the manipulation of input/output devices are most conveniently written in machine language. There are several reasons for this. The more intimate contact with hardware gained by using machine language helps the student to understand exactly what the machine does to the measurement. It frees more core space for data storage and allows maximum efficiency and speed in data sampling. Finally it is easier to match machine language with special-purpose software such as the FFT sub-routine supplied by the manufacturer. Therefore the student lab manual for the course was written so as to provide an introduction to machine language programming, as well as an outline of experimental procedure. (2) The experiment was divided into a series of weekly sub-experiments through which the student learns how to operate peripheral devices by direct command. At the same time insight is gained into the effect of computer sampling on the measurement process.

The teletype is the most important peripheral device since it represents the main control console for a minicomputer. The first experiment teaches the student how to send and receive characters between the teletype and the accumulator register. The initial program is as follows:

```
*200
 START,  CLA     0200    7600
         LAS     0201    7604
         TLS     0202    6046
         HLT     0203    7402
```

The student is given the symbolic listing at the left and must translate it into the machine language version at the right, expressed here in octal notation. The entire program is then toggled into core via the switch register. The program begins at location 0200_8 in the core memory. This location is designated by the code word "START". The first instruction, CLA, clears the accumulator. Then follows the command LAS which causes the accumulator to be loaded with the contents of the switch register. The student must, of course, preset the switch register, or SR, with the desired number or letter in ASCII code. Next the TLS command activates the teletype causing the character in the SR to be printed. Finally a HLT or "halt" command brings the operation to a conclusion. Entire words and even sentences can be laboriously typed out this way by entering ASCII character equivalents one at a time by hand into the switch register.

In the second program the HLT command is replaced by a jump back to "START" so as to make the program operation continuous:

```
*200
START,   CLA          0200    7600
         LAS          0201    7604
         TLS          0202    6046
         TSF          0203    6041
         JMP.-1       0204    5203
         JMP START    0205    5200
```

Here it is necessary to insert a "writing loop" to prevent a pile-up of uncompleted teletype actions. This provides an opportunity to emphasize a basic rule of I/O programming: Allowance must always be made for the slow response of external devices relative to the speed of the central processor. The teletype is a mechanical system and requires about 0.1 sec. to do its thing. This is thousands of times longer than the execution time of the rest of the program. To prevent repeated TLS commands from interfering with each other the instructions:

```
TSF
JMP.-1
```

are inserted. The command TSF senses when the mechanical action has been completed and causes a "skip" of one instruction. Meanwhile the computer keeps repeating the "flag" test TSF. The instruction JMP.-1 means "jump back one step".

The inverse operation whereby a single keyboard character is inputted to the accumulator is accomplished by the following program:

```
*200
START,    KRB          0200   6036
          HLT          0201   7402
          JMP START    0202   5200
```

Here the selected key is typed before the program is started, setting up a store ASCII character in the keyboard buffer register. The command KRB causes the accumulator to be cleared and the contents of the keyboard buffer register to be transferred into the accumulator, whereupon the computer halts. When the operator restarts the program where it stopped the process is repeated. No timing problem occurs here since the program is self-stopping.

Succeeding experiments introduce commands for operating the A/D and D/A converters. The exact timing of the conversion process can be externally controlled by the command SYNC and its associated "flag" test SYSF. These operate a homemade Schmitt trigger control element built up from D.E.C. logic modules. A convenient "clock" is an ordinary audio oscillator whose dial then serves as an indicator of the sampling frequency.

At this point the student is able to employ the system as a digital voltmeter to measure voltages in the range -5 to +5 volts. As an exercise a 1.5 volt dry cell is placed across the input and the measured voltage must be deduced from the binary number found in the accumulator. This involves first translating the number from binary into decimal and then adjusting the result for the calibration factor of the converter. Fullscale reading of 7777_8 (which is the same as 4095_{10}) corresponds to minus five volts at the input, 4000_8 (or 2048_{10}) is zero volts, and 0000 is five volts positive. Thus a -1.5 volt input results in a (decimal) reading of

$$2048 + 1.5 \times \frac{2048}{5.00} = 2662_{10}$$

which, in turn, will appear in the accumulator as the octal number 5146_8.

With the introduction of the D/A command DYS, or "display", the student can cause the reconverted analog of the data being read into the accumulator to simultaneously appear on a scope trace at the same sampling rate. A sine wave input will then give a scope trace output (when properly synchronized by the scope sweep circuit) similar to Figure 3c. This permits the data recording activity to be "monitored", so to speak, by the operator.

2. Data Sequence Storage.

The next group of experiments is designed to develop skill in storing and managing time series data for periodic input signals. The student must first write a program which will store 64 discrete data points in consecutive order, using one core word for each data point. In order to do this he must learn how to "initialize" a data-taking operation. What is involved is 1) assigning a "pointer" location for the initial core address of the data buffer, and 2) setting up a counter location to keep track of the number of data samples.

In Figure 4 is shown a complete program (Symbolic Assembler coded) for sampling exactly one period of a periodic signal. The program is designed to be used interactively. A block diagram of the complete apparatus is given in Figure 5. The signal to be recorded, say a sound from a musical instrument, is played into a microphone/amplifier unit and thence enters the A/D input jack as an electrical voltage. The operator sets the sampling rate oscillator at an appropriate frequency and starts the program. The computer begins to fill the 64-point data buffer region of core with equally-time-spaced data samples. At the same time it outputs a reconstructed signal to oscilloscope and loudspeaker so that the operator can see and hear what the stored version is like. By manually adjusting the sampling rate he can cause the 64-point buffer to account for just one period of the incoming signal. (That is, he sets the sampling rate to 64 times the signal frequency.) When he feels that this adjustment has been correctly made, he touches a teletype key to "freeze" the data buffer. The KSF keyboard flag command senses this action and skips the program into a playback routine which begins at the location designated "PLAY". Playback is accomplished by simply reading out the 64 points over and over again at the original rate. If the sampling rate is not quite right the reproduced sound will differ in frequency and character from the original and the scope trace will be distorted. By a process of successive approximations it is feasible to synchronously sample exactly one period of a signal. This, incidentally, is something virtually impossible to do on a conventional analog-type tape recorder.

3. Function Generating.

In a further development of the same technique the student can fill several data buffers with various signals and then cause each buffer to be sounded in response to a specific teletype key. If the buffers are of varying length it is possible to set up a musical scale in this fashion which can be played from the keyboard like chimes.

Of more practical application is the use of the playable data buffers as function generators for other laboratory experiments. The computer is the only truly universal function generator. From the well-known Sampling Theorem (3) it can be proved that any band-limited signal can be perfectly represented by a finite time series. Arbitrary signal functions can therefore be created by setting up their corresponding time series in buffer or, if available in real time, recording one period by the method described.

4. Aliasing and Information Theory.

The student can now be made to understand the importance of low-pass filtering and the relation of bandwidth to information transmission. The phenomenon of "aliasing" is easily demonstrated by introducing a sinusoidal signal directly into the A/D converter and then raising the signal frequency until it exceeds the sampling rate. As the input reaches one-half the sampling rate, called the "Nyquist" frequency, the sound heard at the output speaker begins to decrease in pitch. The lowering of pitch continues until the sampling rate frequency is reached, after which it starts up again. The phenomenon is periodic for multiples of the sampling rate. By placing a cut-off filter at the Nyquist frequency, the effect is eliminated, however, within the passband, high pitched "noise" will be heard on account of the discrete nature of the analog output. The noise signal consists of the spectrum of the sampling frequency, its multiples, and their "side bands". It can be easily removed by filtering the output at the same cut-off as the input.

With double-filtering, the student can utilize the computer as a high fidelity recording system, up to the limits of data storage capacity. By filling nearly the entire 4096-word core with data it is possible to record one-second messages quite intelligibly. That amount of core permits a data-sampling rate of 4000 Hz, giving a bandwidth after filtering of 2000 Hz, which is about that of a telephone. Examples of student messages preserved this way are "Hello dere", "Merry Christmas", and "Beat Army".

5. Advanced Experiments.

Since the early programs were quite short, the students coded them into machine language by hand. Although this gives valuable and intimate contact with the hardware, it is an inconvenient procedure for programs of greater length. In advanced studies with the computer students were taught how to use a symbolic assembler to prepare binary-coded paper tapes. Also they were encouraged to learn additional programming skills on their own.

A number of useful techniques are explained in an instruction manual supplied by the manufacturer. (4) For example there is an algorithm particularly useful in data taking which is the routine for decoding binary numbers in core into decimal printout on the teletype. Many of the computational advantages of using higher level languages are available in assembler code in the form of manufacturer-furnished subroutines. For the PDP/8 these include a floating-point package, trig functions, and even a Fast Fourier Transform. A student who took the course last semester is currently developing the FFT subroutine into a spectral analyzer program for oscilloscope display.

IV. SUMMARY AND CONCLUSIONS

The high power and low price of the current generation of minicomputers make them an attractive medium for introducing hardware-centered experiments into computer-based courses. Because the minicomputer need not be linked to the large central computing system it is safe to allow students to have hands-on experience in machine language programming. By the addition of an inexpensive analog-to-digital converter, laboratory experiments can be performed which would be impossible on a conventional time-sharing terminal. A good example is real time signal sampling. The further addition of a digital-to-analog converter allows the data-taking process to be monitored on an external oscilloscope creating an interactive measuring tool under the control of the operator. The same configuration can also be used as a universal function generator for other types of experiments.

In conjunction with a computer-based course in acoustics, a two-hour-per-week laboratory was develojed around a series of experiments utilizing a Digital Equipment Corporation model PDP8/I. Students easily developed enough machine language programming skill to perform fundamental measurement operations involving discrete sampling, storing, and displaying of data. Some have been interested enough to follow up the introductory experiments with more advanced data-processing programs.

One difficulty that has been encountered is the fact that the computer console is available to only a few students at a time. When the course is offered again in the Fall it is planned to implement a suggestion of Dr. Arthur Luehrmann of Dartmouth to utilize the large time-sharing computer in assembling programs for the minicomputer. This is an acknowledgement of the many convenient terminals of the big system, which points up the complementarity between large and small scale computer facilities. It's nice to have some of each!

REFERENCES

1. S.A. Elder, "The Naval Academy Acoustics Course: An Experiment in Computer-Oriented Instruction," p. 10.12-10.20, Proceedings of a Conference on Computers in the Undergraduate Curricula, Iowa City, June 1970.
2. S.A. Elder, *Laboratory Manual for Acoustics*, USNA Physics Dept. Lab Manual Series, September 1970.
3. R.W. Hamming, *Numerical Methods for Scientists and Engineers*, McGraw-Hill, 1962; pp. 308-310
4. *Introduction to Programming*, Digital Equipment Corporation, 1969.

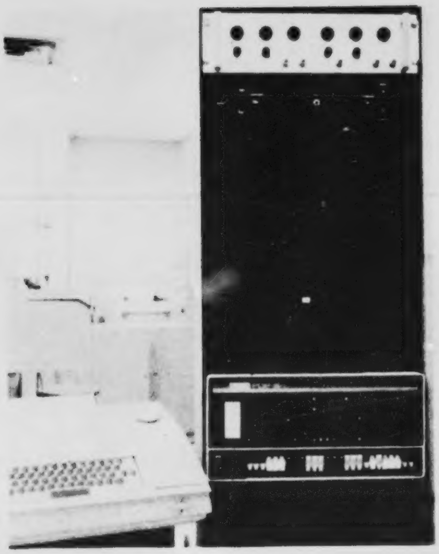


FIGURE 1

PDP8/I COMPUTER CONFIGURATION

The computer console panel is at the lower right. Just above it is a high-speed paper tape reader. At the top of the rack is a dual channel low-pass analog filter set. Below this is a panel containing BNC input, output, and SYNC connectors associated with A/D and D/A conversion.

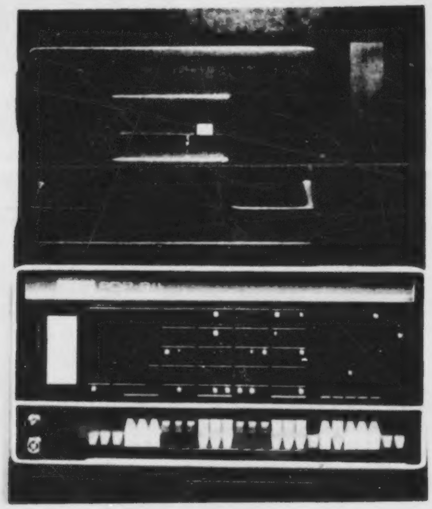


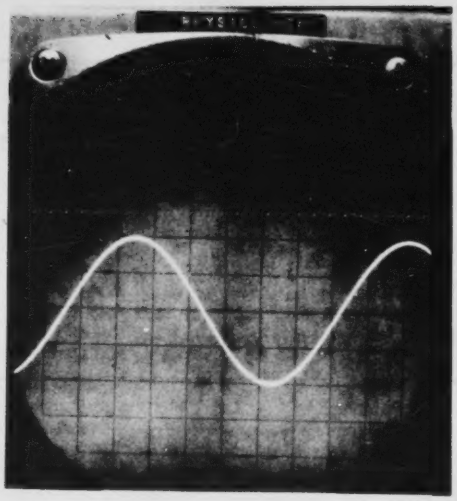
FIGURE 2

COMPUTER CONSOLE PANEL AND HIGH SPEED READER

The twelve rocker switches immediately below the light indicator registers make up the 12-bit switch register of the instrument. Other rocker switches are for extended memory addressing and control purposes.

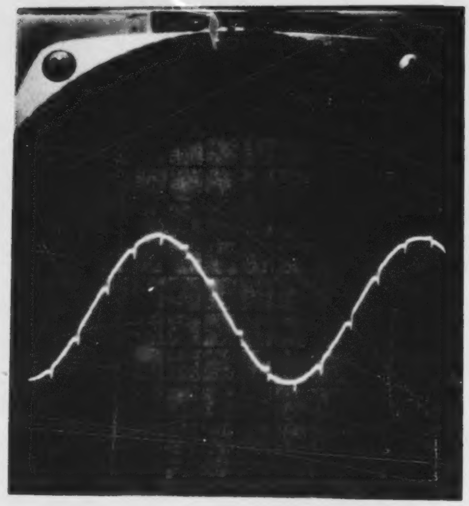
FIGURE 3

A. ANALOG INPUT SIGNAL:
2000 Hz SINE WAVE



B. OUTPUT OF SAMPLE-AND-HOLD AMPLIFIER.

Sampling rate is 20 KHZ. Note 10 microsec "hold" plateaus, spaced every 50 microsec. (Horizontal time scale of oscilloscope is about 60 microsec./div.)



C. OUTPUT OF D/A CONVERTER (BEFORE FILTERING)

Playback rate is 20 KHZ.

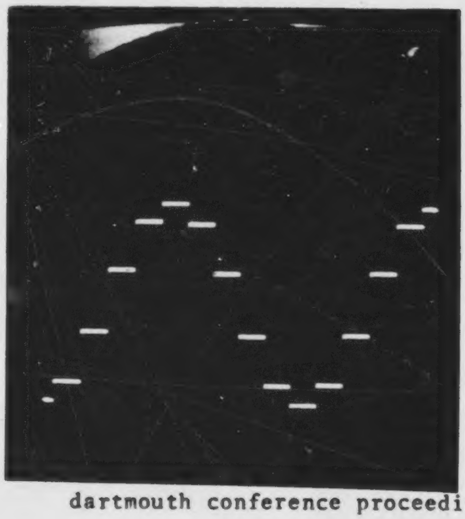


FIGURE 4

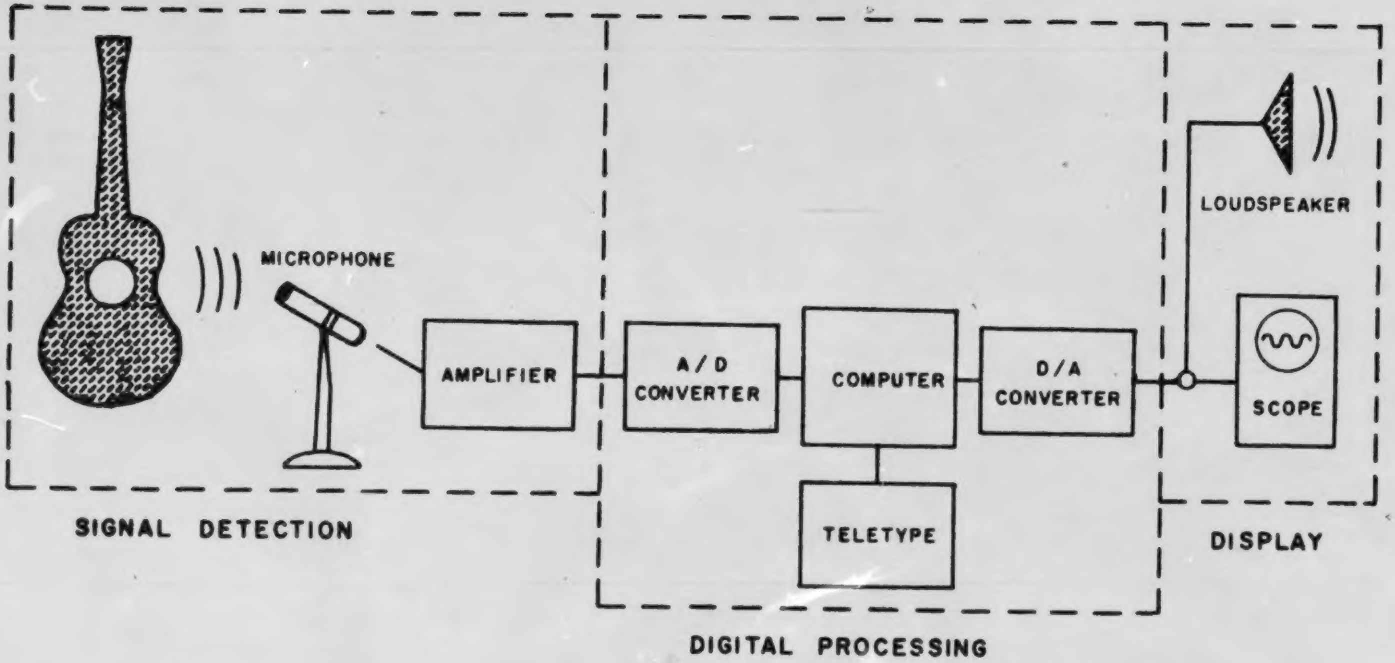
5. FREEZE DATA.

A) PROGRAM TO SAMPLE AND PLAYBACK DATA.

```
*10
PT1,     0
*20
BUFF,    400
PERIOD,  100
MPER,    0
*200
START,   JMS INIT
         SYNC
         SYSF
         JMP.-1
         JMS DATA
         DYS
         DCA I PT1
         ISZ MPER
         JMP START+1
         KSF
         JMP START
PLAY,    JMS INIT
         SYNC
         SYSF
         JMP.-1
         TAD I PT1
         DYS
         CLA
         ISZ MPER
         JMP PLAY+1
         JMP PLAY
INIT,    0
         TAD BUFF
         DCA PT1
         TAD PERIOD
         CIA
         DCA MPER
         JMP I INIT
DATA,    0
         ADCV
         ADSF
         JMP.-1
         ADRB
         CLL RAR
         CLL RAR
         JMP I DATA
```

(1) MAKE A BINARY TAPE OF YOUR PROGRAM WHEN YOU ARE SURE THAT IT IS TOGGLED IN CORRECTLY. FOR THIS YOU NEED THE BINARY PUNCH PROGRAM (SEE WRITEUP).

FIGURE 5


MICROPHONE
AMPLIFIER
A / D CONVERTER
COMPUTER
D / A CONVERTER
TELETYPE
LOUDSPEAKER
SCOPE
SIGNAL DETECTION
DIGITAL PROCESSING
DISPLAY

COMPUTER-AUGMENTED INSTRUCTION IN AN INTRODUCTORY MANAGEMENT STATISTICS COURSE

David P. Doane
Harvey A. Shapiro
Kenneth C. Young
Oakland University
Rochester, Michigan 48063

INTRODUCTION

Oakland University, like many schools located near major metropolitan areas, has experienced rapid growth during the last few years. In the School of Economics and Management, the transition has pointed up a basic dilemma: Can a rigorous program be designed to serve increasing numbers of undergraduate students without sacrificing the flexibility and close student-instructor contact which are highly valued by faculty, students, and administration (and rightly so) of a "once-small" school? In the core courses of the management program at Oakland, some procedures have been worked out which permit a tentative "yes" answer to this question. The judicious use of the computer for instructional purposes is an important part of our present structure for obtaining optimal learning outcomes within the limits of the available budget. Other elements in our strategy include limited use of videotaped instructional lectures and trained undergraduate students as monitors and tutors.

In this paper, we will focus on the teaching of statistics to undergraduate management majors. The course is divided into two parts, a lecture and a laboratory in which extensive use is made of computers in the time-sharing mode. The computer is used not merely as a computational aid but also as an instructional device which is coordinated with the lecture portion of the course to increase the time available for instructors to discuss fundamental issues and concepts.

Fred S. Keller* suggests a list of goals for improving instructional methods which the authors feel may be especially useful in courses designed to produce student mastery of technical tools (such as statistics). Among these goals, Keller includes: (1) allowing students to proceed at their own pace; (2) requiring student mastery of material at each step of the learning process; (3) using lectures to motivate rather than convey critical information; and (4) using student proctors for tutoring and evaluating student performance. These suggestions are useful in determining the extent to which the computer should be used in a curriculum, since it can help in achieving all of these instructional objectives.

To the extent that computers, videotapes, and student monitors serve as a substitute for professorial instruction, there presumably exists an optimal allocation of each type of resource based on its relative cost and on its effectiveness with respect to generating the desired learning outcomes. The system we will describe here is not adduced as a universally applicable solution to this optimization problem. However, insofar as it is largely machine- and textbook-independent and because it contains various user options, it might reasonably find use in programs at other schools of varying sizes. Further, the topics included are virtually universal in beginning management statistics courses across the country.

The goals of the required management statistics courses are (1) to teach students the computational techniques of modern statistics; (2) to develop students' comprehension of statistical testing and of the underlying probability concepts; and (3) to train students in the choice of appropriate techniques, in terms of underlying assumptions of various tests, for the varying classes of data to be analyzed.

The programs in the lab were written entirely by the authors and were designed to achieve the following goals: (1) to make full use of the interactive capability of time-shared terminals in order to permit the user to exercise his own judgment regarding options available in the program; (2) to perform useful statistical calculations; and (3) to give feedback to the user in the form of warnings about (statistically) undesirable outcomes associated with his decisions. Since these programs are written in BASIC, they are largely machine-independent. However, their pedagogical value in a batch mode system would be severely diminished.

The course is divided in a conventional manner into four parts: (1) descriptive statistics; (2) probability and probability distributions; (3) sample distributions, estimation and statistical inference; and (4) correlation, regression, and time-series analysis.

PROGRAMS AND COURSE OUTLINE

Descriptive Statistics

In the descriptive statistics portion of the course, students are introduced to four types of computational techniques: (1) use of simple calculators; (2) use of programmable electronic calculators; (3) use of "canned" computer programs; and (4) computational algorithms and flowcharting for computers.

The two computer programs used in the descriptive section have been deliberately kept simple in order to facilitate the explanation of their algorithms to students and to impose on them the requirement that they use a certain amount of judgment in their use.

The students are introduced to the use of the computer time-sharing terminal through HISTO, a program which constructs a frequency distribution from raw data and prints out a crude histogram. In addition to their input of the raw data, students are required to choose the lower limit of the frequency distribution and the interval size. Since they use time-sharing terminals, the students are encouraged to experiment with different limits and class sizes. This is intended to encourage them to view the computer as simply a complex calculator under their control. Along with the other output, a warning is printed if the number of intervals used differs greatly from the number suggested by Sturges' rule. This permits a student to revise his earlier decision, if he desires.

The second program ANLYZ, calculates a variety of descriptive statistics from a vector of raw data. This program is used to introduce students to the power and speed of the computer in complex calculations (e.g., the fourth moment) and to impress on them the importance of understanding the precise algorithms used in a "canned" program. For example, sine the program does not state which of various possible methods are used to calculate the "coefficient of skewness", students must do a bit of detective work in order to interpret their output.

Calculators and Computational Algorithms

In this course, no effort is made to teach computer programming languages. However, it is desirable to acquaint the student to a limited extent with the internal workings of the computer rather than to treat it as a complete "black box". To this end, a taped lecture is provided on the workings of a less complex machine (the Monroe EPIC 3000), a programmable calculator. The ideas of input, output, storage, and sequential arithmetic operations are thus illustrated without introducing programming languages explicitly.

This calculator has five memory registers and may be programmed to perform up to 42 separate operations on data. Computational algorithms for doing simple statistical calculations are demonstrated.

The next lab exercise requires the students to make a flowchart incorporating an algorithm for calculating some statistic they have already studied in the lecture. This bridges the gap between simple calculators and a full-size computer. Students can see the need for a more elaborate language to execute their algorithms without actually writing a computer program.

Sampling Distributions

Since a familiarity with the behavior of sample statistics is essential to an understanding of inferential statistics and is often a difficult concept for students to grasp, the computer is utilized as an instructional device to supplement the lectures and problems.

The first program, DICE, introduces students to the notion of sums of random variables. The computer program, utilizing a random-number generator, simulates the throw of one, two, or three dice. The program accumulates the results of successive "throws of the dice" and produces, on command, a histogram of each of the three distributions. To facilitate this comparison, class interval sizes are adjusted so that each histogram has six classes. For the throw of a single die, each class simply represents the number of times a particular face appears. For the pair of dice, the class interval size is two, and for three dice the interval size is three. Operation of this program gives students direct, empirical confirmation of classroom proofs that sums of a uniform random variable have a symmetrical binomial distribution.

A second program, MONTE, uses Monte Carlo simulation techniques to generate distributions of sample means drawn from empirical frequency distributions of the user's choice. The student's input consists of class frequencies for each of nine intervals (the number of observations is arbitrarily set at 90). At the beginning of the run, the computer generates 20 samples of size 10, prints out each mean as it is computed, and then prints a histogram of the distribution of sample means along with a histogram of the distribution of the population.

At the user's option, additional sets of twenty sample means are generated and the histogram of the accumulated distribution of means is printed. This can be repeated up to five times. By initializing the program with differently shaped distributions, the student can not only develop an intuitive feel for the meaning of the central limit theorem but can also acquire some sense of the rates of convergence of sample means toward a normal distribution when drawing from different types of populations.

Chi-Square

The third program used in the inferential statistics part of the course is a simple chi-square test adapted to our pedagogical purposes. In beginning statistics courses where computers are used, a good deal of attention is ordinarily focused on regression and other parametric methods for measuring association among variables and for testing hypotheses. This is natural, for such exercises nicely illustrate the power of the computer as a labor-saving device. It is not so common in beginning courses to worry extensively about the many assumptions presupposed by these techniques (shapes of the distributions of residuals, randomness of variables, etc.). Often, certain non-parametric methods (such as chi-square) will suffice to check for independence of variates or for goodness-of-fit. Non-parametric methods are especially appropriate for the class of management problems where data are ordinal numbers (e.g., yes/no answers or information on socioeconomic background). In many cases, a simple contingency table can add to our information without precluding other more elaborate tests if they are warranted. In this course, we explicitly point out the principle of Occam's Razor: given two techniques, either of which is sufficient to resolve an empirical question, the student should prefer the technique requiring fewer assumptions.

For tests of independence, our time-sharing program CHISQ accepts up to 100 observations on each of two variables. The high, low, and range of variables are printed. The user is then asked to specify the number of intervals desired for each variable in the contingency table (up to 5x5). The computer prints out the contingency table (along with interval labeling information), the chi-square test statistic, and degrees of freedom. If a correction factor has been used, this is stated along with a warning if any class does not contain a sufficient number of observations. Here, the value of on-line work becomes evident. The user may now alter the number of classes for either variable and obtain a new contingency table. The student is warned of his mistakes, but he is left to judge how to correct them without having to re-enter his data.

The goodness-of-fit version of CHISQ simply calculates a test statistic from the user's data on actual and expected frequencies under various assumptions of his choice. A warning is printed along with the output if the specified number of intervals exceeds the number suggested by Sturges' rule. The advisable number of intervals is printed, in case the user wants to follow through.

Regression and Time-Series Analysis

Although a great many "canned" programs exist for regression and time-series analysis, these seem to the authors to be poorly designed for their pedagogical purposes. Since a major goal of the statistics course is to instill in students a wariness of the inappropriate use of linear regression techniques, it was felt desirable to introduce students to tests on the residuals of regression analysis. Hence, standard computer routines have been modified to feature tests of the assumptions as a prominent part of the printout. This immediate feedback is especially useful when (as frequently happens) the data fail to satisfy one or more of the requirements at the levels of significance the user himself has selected.

Three tests are performed on the residuals in connection with each linear regression run by the students. They are: (1) a chi-square test for a normally shaped distribution; (2) Bartlett's test for heteroscedasticity; and (3) the Durbin-Watson test for autocorrelation. Although these forbidding sounding tests are normally reserved for advanced courses, it is possible to explain their rationale in terms of elementary concepts which are covered in the introductory course. The chi-square test for normality is, in fact, the last major topic in the statistics course prior to using regression analysis.

Two similar computer programs are used for this section of the laboratory course, BIGRES and MGRES, the first for bivariate regression and the second for multivariate regression.

Each program requires the students to enter, in addition to the observations for each variable, critical values for each test. Thus they are forced explicitly to consider the assumptions underlying the technique each time they run the program. Output of the test statistics, along with a verbal interpretation of their meaning, is printed before the usual output of regression programs.

The first test on the residuals is a simple chi-square goodness-of fit test. The computer program groups the residuals into class intervals, using Sturges' rule for the number of classes. It then computes the expected number of observations in each class for a normal distribution with the same mean and variance. The computer then calculates the chi-square test statistic and compares it with the critical value which has been entered by the student, and prints a warning if the hypothesis of normality can be rejected.

For our purposes, an arbitrarily restricted form of Bartlett's test is employed which can be performed automatically by the computer. The residuals are divided into two subsets based on the ranking of the first independent variable. In this form, Bartlett's test is a test of the hypothesis that the difference between the variances of the two subsamples is the result of sample variation. This is a familiar concept; and, since it also uses the chi-square tables, is easy for the students to apply. Again, the computer prints a warning if the null hypothesis can be rejected. The Durbin-Watson test statistic is interpreted as an approximation of the coefficient of correlation between successive residuals. This test does require the use of a table which is new to the students at this point in the course. However, the computer program takes the critical values entered by the students, does the fairly elaborate computation required by this test to ascertain the regions of acceptance or rejection, and prints the outcome.

LAB FORMAT

A total of seven assignments is given for the lab section during the semester. Each assignment requires the student to gain access to the time-sharing system, call out one of the "canned" programs described earlier, and enter a set of relevant data. Each student is provided with a usage number and a password which allow him to access an off-campus (commercial) time-sharing system. Connection to the system at present is made via three terminals, each of which is available to the students 43 hours per week. Enrollment in the lab runs about 100 students. During the hours of terminal availability, a trained student monitor is always available to answer questions concerning computer problems.

An introductory lecture is provided before each program is used. At this time, a write-up is distributed which includes instructions for the use of the program and a sample dialogue from a typical run. The lecture sections meet twice a week, each session lasting approximately two hours. The lab meets once a week for one hour. Two identical lab sections are conducted at different times and students are allowed to attend either session.

Due to the repetition involved in presenting a lab lecture twice, the instructor arranged to have the first session videotaped. The second session is then a replay of the tape. All tapes are preserved so that at any time a student is able to view a lecture that he missed or failed to understand. This feature is particularly useful during the first part of the semester to those students who sign up for the course late and miss a lecture or two.

The grading of the lab assignments is +,-, or 0 (satisfactory, unsatisfactory, or not turned in, respectively). To achieve a perfect grade in the course (4.0), a student is required to get seven pluses. Assignments can be turned in any time until a cutoff point a week before the end of the semester. If a student receives a minus, he is permitted to redo the assignment and resubmit it. In this manner, each student has perfect information concerning grading and can achieve any grade he desires by resubmitting assignments until he has the "right" number of pluses.

STUDENT REACTIONS

Early in the Winter 1971 semester, the students who attended the video lab section were asked to critique the usefulness of the videotapes. Their comments mainly concerned the quality of the video and audio presentation in terms of technical problems. No students complained about not getting a "live" lecture.

The problems encountered while videotaping involved the loss of audience participation. The questions asked by the class during the taping were poorly recorded. This was due to the lack of a "shotgun" (highly directional) microphone which can effectively pick up student questions. The instructor, therefore, attempted to repeat all questions before giving an answer.

An interesting option that was sometimes exercised was to stop the videotape to clarify points which the instructor felt were not adequately explained on the tape even though no questions had been raised. The students mentioned this interaction in their critiques and expressed the view that the presence of the instructor was necessary to guarantee a full understanding of the subject matter. One student's comment came as a surprise: "Videotape lectures eliminate any distractions which may otherwise take away my attention momentarily in a live lecture." The authors feel that this can be attributed to the fact that students are watching only a small area; they therefore need to move their eyes fewer times and do not notice movements occurring on the periphery of their vision. Also, the room is kept somewhat darker during playback to prevent reflection of the ceiling lights on the television screens. This could also account for fewer instances of distraction by the viewers.

The reaction to the organization of the lab assignments has been quite favorable also. Students like the option of repeating an assignment until they earn a plus. In addition, they appreciate the absence of due dates. This freedom from time pressure is, we believe, an important motivational device and produces high-quality work. In most cases, instantaneous feedback is given to the students; even though the class sizes have been large, the instructor records a message on any unsatisfactory assignment explaining the deficiency. Very few assignments have to be submitted more than a second time before they receive a plus grade.

CONCLUSION

The use of computers, videotaped instructional lectures in the lab, and the use of student monitors in the teletype room are complementary features in the system of instruction. If taken singly or if separated from the lecture portion of the course, these elements become mere aspects of a technology which does not guarantee either cost savings or learning enrichment. It is the consensus of the authors that the educational gains from a "technically augmented" statistics course, such as the one described here, can be analyzed only within a general systems context; the whole cannot be understood by examining each of its parts because of the interactions and reinforcements among various elements of the system. Let us summarize the gains, first from the students' viewpoint and then from the instructors'.

With no lab deadlines, repeatable assignments, and ample access to computing equipment, the student may proceed at a pace consistent with his capabilities. The videotaped lab instructions aid in giving the student maximum flexibility with regard to time allocation. Students' lab work (especially the dynamic element inherent in on-line computer use) allows greater progress, not merely because it speeds computation but also because it illustrates ideas by example in a way that classroom instruction cannot. This latter element is too often missing in a "canned" program lab environment where the emphasis is on merely teaching students to use programs which are scientific, result-oriented packages rather than pedagogical devices designed to aid learning. We, therefore, are arguing in general for statistical program packages which are more student-oriented.

The student can frequently do his homework on the computer and can utilize original data in many cases. He gains a feeling of competence in making his own decisions through the options built into the programs. His powers of judgment are freed from computation so that he may focus on questions of interpretation and experimental design. Finally, students with problems of a minor nature may consult the monitors instead of spending class time on them.

The instructor is freed from playing certain roles, probably those he dislikes the most and which have doubtful educational value. Because he can videotape information of a routine nature, such as user instructions for lab exercises, he needs to repeat things less often, since tapes can be viewed more than once by students who are having difficulty. He can refer minor questions to the lab monitors. He can spend less time at the blackboard illustrating concepts such as histograms, random drawings, and regressions. He can spend more time helping students interpret actual experimental results from the lab and on testing and evaluation of student performance. If a good textbook is used, it is questionable whether the classroom activity can properly be termed a "lecture section" at all.

The cost for the functions outlined above is approximately $1.50 per student semester hour of credit. This is determined by an estimate of two connect hours of time-sharing time (at an average cost of $3.00 per hour) per student added to a total cost for a semester of about $160.00 ($1.50 per student) for the use of the videotape recording equipment. The total cost per student is then $7.50. The reason for the low cost of time-sharing usage is that we have been able to obtain very cheap non-prime time ($2.00 per hour) and we direct the students to minimize their prime-time usage.

While cost savings are difficult to evaluate at this stage, it is our impression that somewhat larger than normal class sizes are acceptable for the lecture and lab sections alike. The use of trained undergraduate lab monitors reduces supervision costs without apparent loss in teaching effectiveness.

FOOTNOTES

* Keller, Fred S., "Goodbye, Teacher," *Journal of Applied Behavior Analysis*, Vol. 1, No. 1(Spring 1968), p. 83.

COMPUTER-PACED INSTRUCTION IN THE BUSINESS CURRICULUM

Thomas J. Dudley and Claude T. Ware
School of Business
Pepperdine University
Los Angeles, California

Background and Justification of Need

In the never ending search to design and implement the most beneficial delivery system for education at the undergraduate level in schools of business, the role of the computer cannot be denied. As a matter of fact, only because of the computer are we, at Pepperdine University, able to begin delivery of an integrated and interdisciplinary approach to business school curriculum.

This paper reports on the experiences of the many who are involved in the planning, implementation, evaluation, and redevelopment of this program which began one and a half years ago. The method employed is a historical view of the situation with some philosophical comment and description of the present process. Indications of future directions conclude this paper.

Historical Setting

Ever since the Gordon and Howell (1), and Pearson (2) reports burst on the scene admonishing schools of business for their lack of quantitative methods and mathematics in their curricula, the trend toward stuffing the student with this kind of material has been increasing. In almost every course today one can find some mathematical concepts and/or calculational techniques as part of the course material. Too often the material is fragmented, incomplete, and inaccurate. In many instances the same concepts are present in many different courses. For example, the mathematical concepts of straight lines are taught in accounting, finance, marketing, economics, etc. The "ideas" of linear relationships are valid in these areas, but the introduction and teaching of these basic concepts should be approached only once and not many times in different courses at varying levels of the total curriculum.

Based on a recognition of this problem a series of two courses was designed to form the foundation of quantitative and mathematical concepts required by students for upper level courses. These two courses, appropriately entitled Foundations I and II, have as content all of the basic quantitative and mathematical concepts. The "list" of concepts was created by the faculty who were involved and confronted with the problem as outlined above. Thus, through their contributions and participation, there was formulated an integrated and interdisciplinary approach to creating course content applicable to the foundation for the total business curriculum.

Delivering Learning Concepts

Having settled on this, however, only led to other problems. How does one deliver these concepts in a meaningful way to all students? Given that students learn, especially mathematics, at different rates, and that all students entering the school must take these courses in their first two trimesters, it was obvious that a new approach had to be created. An end result is a marriage between what has been called "programmed learning" and "computer managed instruction". We would like to propose a new title for this marriage; namely, "Computer Paced Instruction". Prior to an actual discussion of computer paced instruction, however, we probably should at least touch on what we mean by programmed learning. Some comments on the philosophy of programmed learning follow.

The Philosophy of Programmed Learning

In current teaching machine techniques for computer assisted instruction and programmed texts, the arrangement of material follows essentially this pattern:

311

- the learner reads or is told in a concise, brief statement of the learning goal he is expected to achieve;
- he reads a program concept statement (the smallest segment which is considered feasible to master at one sitting) which describes the objectives;
- he is asked to read some material, listen to a tape, view some films, or gather data from informed people on the explanation and application of the learning objective;
- he is asked to respond to some questions based on that information by marking one of the multiple choice answers or writing an answer in a blank space; and
- the correct answer is revealed immediately adjacent to the question or grouped with a small number of other answers a page or two from the question.

He then moves on to the next bit of information, on which he must again immediately answer a question. Classically, each step follows from and builds on the preceding step. The LeMot style of programmed material (3), which is used in writing Foundations I and II course material, is an adaption of the above description in that the smallest lesson unit is a single one-page learning program consisting of a concept statement, program objective, learning activity, and two or more self-assessment problems. This lets the student know he understands each program objective and the manner in which a group of detailed program objectives compose criteria for the achievement of larger increments of chapter and section concept learning objectives.

Preparing Learning Materials

The psychological basis of programming is that a learning task is organized into short sequential steps and that the learner is "reinforced" by successful performance of each program and lesson step. The student is kept informed in step-by-step fashion of his successes and weaknesses in a short period of time. In traditional teaching, however, much time may elapse between the lecture, student readings, the merging of concepts with problem analysis, the testing, and test results. With the programmed material, the learner has the opportunity to check his learning in smaller increments between the learned objectives and the correct responses immediately. By this means the right response is reinforced, wrong objectives are eliminated, and misconceptions cannot creep into the student's learning pattern.

The quality of a particular set of programmed learning materials is not determined a priori. Its effectiveness is tested by the consequences of the use of the materials by students, by student-measured reactions (amount of knowledge, attitudes, and retention time) and evaluations by teachers who use the materials. All teaching materials and methods can be improved by adequate feedback, but programs have a peculiar advantage in that feedback can pinpoint the location and character of teaching and teaching materials' faults.

Managing Students' and Instructors' Time

Programmed learning, because it requires continual responding, ensures that each student is active by his contributions at charted times. Each student progresses at his own rate of speed through the material. The fast student is not held back with the average, and the slow student is not self-conscious; he directs his own learning speed.

Programmed material can relieve the teacher from repetitive aspects of teaching. The teaching machine or computer can present diagnostic, pre-test, and post-test questions, arrange and present learning materials, and automatically score student tests and calculate student and learning material statistical data. A quick and precise method for isolating and correcting problems of student learning, program, design, and administration variables is also afforded by this method.

This in essence was the approach taken to create the course content for Foundations I and II. The function of delivering the necessary feedback and reinforcement so that each student could learn the material in a manner timely to him was designed for the computer. Programmed learning and computer managed instruction form the basis of computer paced instruction.

Computer-Paced Instruction

Definitions

Computer-paced instruction is an educational delivery system in which the students' levels of achievement and progress toward pre-established goals are paced by a computer.

The Foundations course curriculum has been designed and structured along the lines of relevant major subject matter concepts. These major concepts have been subdivided into smaller sub-concept units. These sub-concept units have been related to written material, textbooks, and exercises which explain and require practice to give in-depth meaning to the sub-concepts. The student task is to achieve a level of mastery of all the major concepts within a specified total time period. The student may complete the entire course at a rate of speed that suits him and in the manner that fits his particular method of acquiring knowledge.

The Role of the Computer

This educational experience is called "Computer-Paced Instruction" because of the role which is played by the computer. All of the pre-test and post-test assessments are generated by the computer. The scoring of student performance is accomplished by the computer. The record of what and how the student has achieved, level-wise and time-wise, is maintained by the computer. All the test question items and the educational and management controls for student progress have been programmed into the computer. The one mandatory requirement for continuation through the course is successful completion of the major subject matter concept sections in the sequence in which they are presented. Before embarking on the study of any major concept the student takes a pre-test assessment of question items about each sub-concept within the major concept. If a 90% grade is obtained, the student may go on to the next major concept. If not, he studies the material relevant to this concept in an independent and tutorial manner. When the student feels confident that he has gained the required knowledge, he then takes the post-test assessment on this material. By gaining a 70% or better grade on the post-test he can then move on to the next major concept in the course sequence.

Present Individualized Learning Process

When a student desires to begin one of the major concept sections, he must present himself to the computer laboratory (see Figure A and Table I). He "signs on" the computer and requests the appropriate pre-test assessment. The computer first checks his past performance record to see if he is eligible to take this particular pre-test assessment. If he has satisfactorily achieved the level necessary for him to proceed with this major concept, the computer will so indicate and prepare for him a pre-test assessment.

The pre-test assessment is created in a random manner. On a disc of the computer, some 100 question items relative to this major concept have been stored. Forty of these are then "picked" via the operation of a random number generator by the computer. These forty items are then printed out on the high speed printer. This provides each student with an individualized assessment instrument. The test is taken off the printer and given to the student. He answers the questions at a desk in the computer laboratory at his leisure. Upon completion, the answers are key-punched and fed back into the computer with the student's identification. The computer grades the test, records the results, and reprints the items missed for the student's review.

At specified time periods or upon request, the professor is given certain management reports about this course. These reports are of two general classes. They consist of information about the students' progress, individually and/or collectively. Or, they consist of information about the testing items, also individually and/or collectively. In this way the professor obtains all information necessary to properly manage, adminster, and influence the learning experience. The total educational experience is computer paced because the student must, by course design, interact with the computer periodically. He is forced, in a sense, to determine when that interaction will be encountered; and the computer dictates the actual pace of the accomplishment.

PRESENT INDIVIDUALIZED LEARNING PROCESS

EDUCATIONAL PHASE

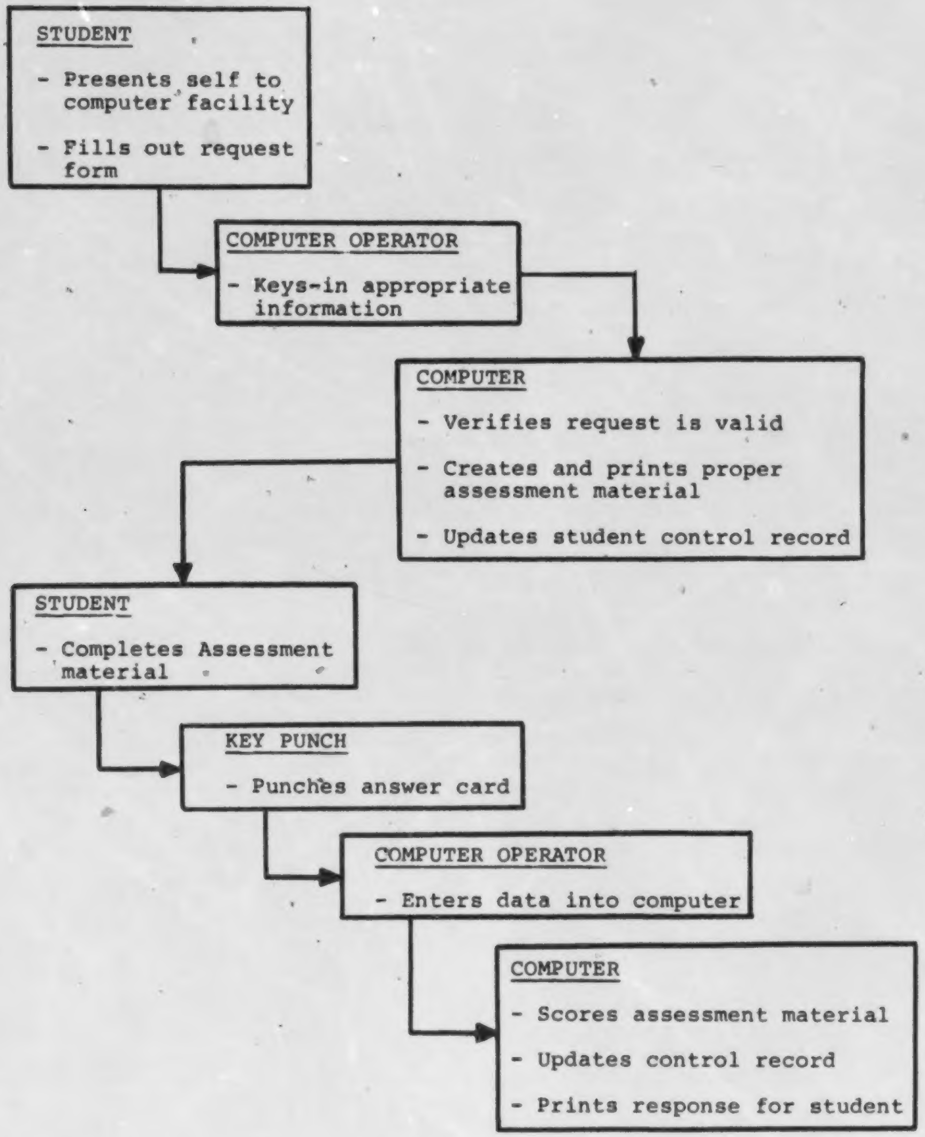

STUDENT
- Presents self to computer facility
- Fills out request form
COMPUTER OPERATOR
- Keys-in appropriate information
COMPUTER
- Verifies request is valid
- Creates and prints proper assessment material
- Updates student control record
STUDENT
- Completes Assessment material
KEY PUNCH
- Punches answer card
COMPUTER OPERATOR
- Enters data into computer
COMPUTER
- Scores assessment material
- Updates control record
- Prints response for student

FIGURE A.

TABLE I

MANAGEMENT PHASE

COMPUTER CREATES

A. According to pre-established schedule:

-Reports on students' progress and achievement
-Reports on assessment items; i.e., test item analyses

B. On command:

-Reports about an individual student's progress and achievement
-Reports about an individual assessment item or group of items

Advantages

There are three major advantages of this type of educational delivery system. Of utmost importance, and contrary to popular belief, computer paced instruction humanizes the total educational experience of the student. It facilitates individual learning at the student's own pace. That is, the student obtains the material at a rate of speed consistent with his ability and within limits according to the amount of time he requires to obtain the knowledge. The amount of time required to obtain a level of achievement becomes the variable. In most present educational situations, the variable is the level of achievement. The amount of time is fixed for all students. With this kind of delivery system all students are afforded the opportunity to achieve the same level of knowledge.

The second advantage is afforded the professor. Because of the automatic management control built into this system, the job of counseling and giving individual attention to students can be particularized. That is, those students who require more assistance will be identified via the computer as time passes instead of in the classroom by his having asked questions which caused him discomfort and wasted class time or by his having flunked a test. Both of these methods, which are presently the mode, are inefficient, uneconomical, and very unfair. By the time the professor learns of a student's problem, it is usually too late to do anything constructive or positive about the situation.

The third advantage of computer paced instruction is of interest to the administration, as it is a financial advantage. With every element taken into consideration the relative costs of operating on-line computer systems per student is less than the operation of present-day educational systems. First, there is the opportunity cost of trading high cost for lower cost elements, e.g. professor and large rooms for graduate student tutors and smaller rooms. Secondly, there is the matter of fuller utilization of the elements. For example, assume that twenty students use five terminals for one hour per day per student; then four terminal/computer hours per day would be utilized. Under present systems of education all twenty students would meet in a room for only one hour of facility utilization per day. In addition, the computer approach requires no professor, whereas the classroom approach does.

Summary and Conclusions

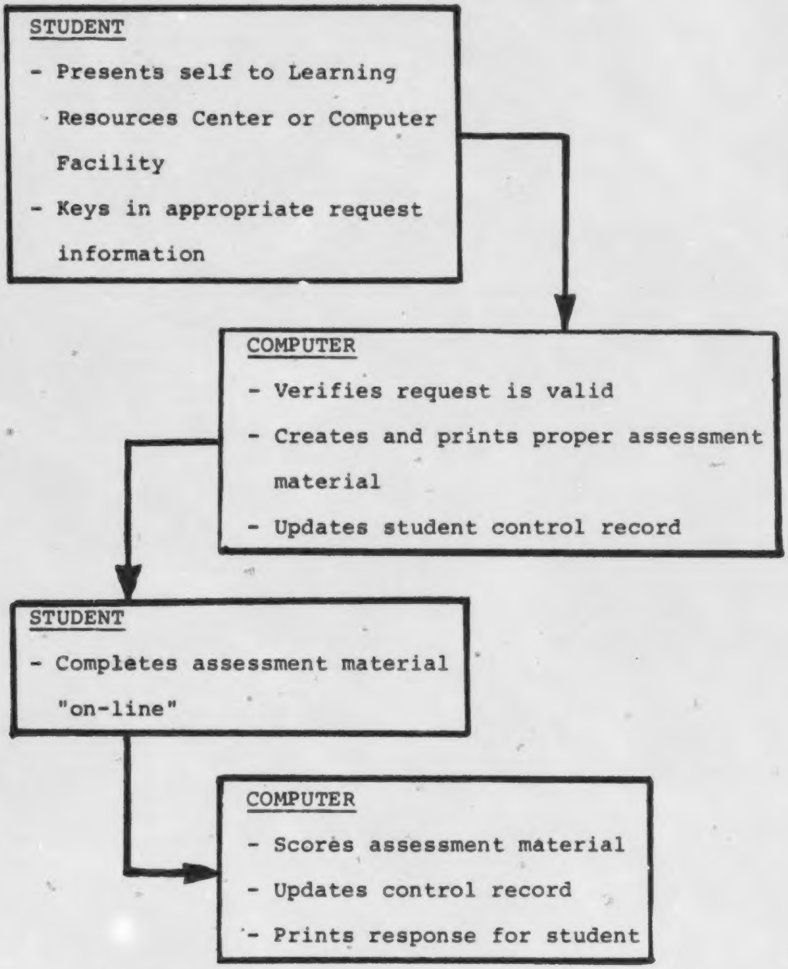
In this (perhaps too brief) explanation of Computer-Paced Instruction, some significant points have been made. First, a definition of this as a flexible, meaningful, and relevant educational delivery system was advanced. Then the role of the computer was spelled out in some detail in regard to the application of this system to a particular course, namely Foundations I and II. Finally, basic advantages to the students, professors, and the administrators were pointed out.

Future Development and Applications

The future for this activity in all likelihood will consist of two different thrusts. The first will be in the area of refining what is presently being accomplished. That is, by continual monitoring of the management reports now, more discriminatory testing items can be

PROPOSED INDIVIDUALIZED LEARNING PROCESS

EDUCATIONAL PHASE



STUDENT

- Presents self to Learning Resources Center or Computer Facility
- Keys in appropriate request information

COMPUTER

- Verifies request is valid
- Creates and prints proper assessment material
- Updates student control record

STUDENT

- Completes assessment material "on-line"

COMPUTER

- Scores assessment material
- Updates control record
- Prints response for student

FIGURE B.

developed. By "beefing-up" the computer system we can implement yet a greater degree of availability to the testing materials via remote terminals. In fact, changes are already in the process of being implemented. Upon completion, the system will function like the schematic in Figure B and Table I.

However, the truly exciting, and in our estimation, really significant future activity rests in the area of concept delivery which this system makes possible. That is, once we have created, designed, and implemented this basic system as presented here, we can then turn our attention to the problem of creating more meaningful experiences for getting the concepts across to the students. What we envisage is a conglomerate of hardware, software, and people experiences put together with one objective in mind: to make it possible for any and all students to understand the concepts which make up the Foundations course. What we have in mind has been aptly described in an article by Theodor H. Nelson under the term "hyper-media" (4).

The student and his achievement should be the central issue. Through "hyper-media" he can obtain whatever experience he requires to achieve the knowledge that is expected of him. The choice of relevant material is his. The total learning experience is complete.

The computer--and thus Computer-Paced Instruction--makes it possible.

Prologue of Acknowledgements

The time and efforts involved in bringing these foundation courses into being were very extensive. The people most directly involved must be acknowledged because without the joint effort of these many, nothing would have been accomplished. First, there was the active and inspirational leadership of the Dean of the School of Business, Dr. Donald R. Sime. The spearheading of the redesigning of the undergraduate curriculum rested with Mr. James Penrod. Much of the actual writing of the programmed learning material was accomplished by Mr. Randall Stephens. The computer programming and operations was conducted by Mr. Paul Ray and Mr. Mike Anastasion. Producing and general coordination rested with Mr. David Ralph. The University, in general, and we the authors, in particular, owe many thanks to these people.

REFERENCES

1. R. A. Gordon and J. E. Howell, Higher Education for Business, New York: Columbia University Press, 1959.

2. F. C. Pierson and Others, The Education of American Businessmen, New York: McGraw-Hill Book Co., Inc. 1959.

3. Hillery Motsinger, LeMot Instructional Packages, Pepperdine University, 1969.

4. Theodor H. Nelson, "No More Teacher's Dirty Looks", Computer Decisions, September, 1970.

THE TIME SHARING COMPUTER AS A MANAGEMENT SCIENCE LABORATORY

J. Ronald Frazer
Clarkson College of Technology
Potsdam, New York 13676

A time sharing computer terminal has great potential not only for computational work but as a source of data for laboratory experiments and simulation of management problems. The basis for the data lies in the random number generation techniques developed on the computer primarily for Monte Carlo simulation. However, while the techniques used are similar to those developed for Monte Carlo work, the philosophy and the applications are quite different.

In Monte Carlo simulation the typical practice has been to run exhaustive tests to establish a particular parameter within quite narrow limits or to convincingly disprove or confirm a particular hypothesis. The emphasis has usually been on solving the problem rather than on the inherent statistical question of how much information can be secured from a limited amount of data. Thus, computer simulation techniques have tended to be of most interest to practitioners of operations research as contrasted with those most interested in management decision making, where all too often it is necessary to make decisions with relatively little information.

While teaching Monte Carlo simulation in an operations research course, it is not uncommon to use the technique of assigning individual students in a class the responsibility of getting a simulation program running to solve part of a problem, putting the results of all student programs together to secure an overall answer to a project. One example of this would be in simulating an inventory model where the individual students might be assigned different reorder points to test, with the results of all student programs being used to secure an estimate of the optimum reorder point.

As part of this process, one necessarily becomes involved in the question of whether all students have programs that are appropriately simulating the inventory model under consideration, and this fairly naturally leads to applying simple statistical tests to establish any significant deviation from the expected results. From an educational viewpoint, the impressive thing about doing this is to observe how amazed many competent students are to find that statistics really works. These students have been well trained in statistical procedures, calculations and tests but have not really made application of these procedures a part of their tools of the trade. Using the inventory simulation as an example, let us assume that each student simulates for 10,000 days a demand generating daily usage as a normal distribution with mean = 50 and standard deviation = 10, that the Order Quantity used was 500 and the ordering cost $25. One might find that student data on ordering costs for 10,000 days was as follows:

Student	Ordering Cost
A	25,075
B	24,975
C	25,025
D	24,800
E	24,975
F	24,975
G	2,500
H	25,100
I	25,050
J	24,700
K	25,025
L	24,925
M	25,025
N	24,925
O	24,775

On receiving this data virtually all students will recognize that student G has an invalid program, but look on it as black magic when the instructor says something like "In addition to Student G, Students D, J, and O have invalid programs and very likely made the same mistake, leading one to believe they may have worked together in developing the program." It helps to demonstrate that the standard deviation of usage for 10,000 days is $10 \times \sqrt{10000} = 1000$, and thus the standard deviation of order cost is 1000 x ($25/500) = $50, but the fact that statistics really works is not usually fully comprehended until the error in the program is found and corrected. The error in this case can fairly readily be detected to be a rounding error in programming, resulting in the mean demand being 49.5 rather than 50.0. Correcting the error and rerunning demonstrates that results will fall in the ± 3 sigma range of $24,850 to $25,150.

Experiences of this sort led to using the computer and Monte Carlo type data as a device to teach statistics, where the emphasis is on analyzing the available data, rather than as a device solely to get answers to particular questions. In doing this, one quickly becomes impressed with the great improvement in efficiency of learning that can be secured by using a time sharing computer terminal with immediate response rather than the typical long-wait, batch type computer operation. I began developing programs designed to provide an opportunity to use several different statistical tools using a batch computer about three years ago, then shifted to a time sharing RAX system that was less than satisfactory, and went from there to using GENERAL ELECTRIC's MARK II service, programming primarily in time sharing FORTRAN, and this has proved quite satisfactory.

Several programs have been developed at different levels ranging from elementary programs designed to illustrate basic statistical concepts such as the central limit theorem to considerably more sophisticated simulations in which the student is expected to compete with the forces of nature--as represented by the computer--or with other students in a joint simulation.

These programs can be designed to illustrate many different statistical and other relationships. In addition, they can be used as takeoff points for further analysis and discussion, with the scope being determined by the level of the student, the time available, and the type of course (whether statistics, financial management, or operations research, etc.). No one program can do any more than be illustrative of the many things that can be done. However, to illustrate, I would like to describe a relatively simple competitive simulation called "Stock Market". The name is used with some misgiving because no actual stock market is being simulated. Rather what we are doing is setting a statistical inference problem in a competitive bidding situation, where the competitive bidding and need for management of resources in some measure simulates this part of actual portfolio management.

The rules of the simulation are detailed in the following set of instructions given to the students:

Stock Market

This simulation is called a stock market simulation because it attempts to simulate the action of investors in the marketplace where the true long run value of any stock is uncertain and subject to individual appraisal. It is designed so that the emphasis will tend to be on trading--hopefully buying at the low and selling at the high--rather than sitting back and collecting dividends. There is no simulation of the analysis necessary to select one stock in preference to another in the usual sense. Instead, pure statistical inference is substituted for the typical investment analysis.

There will usually be five companies in the simulation and from six to ten investors (individuals or teams). Each investor will begin with 1,000 shares of stock in each company, with each share having an initial value of $100. In addition, he will have $50,000 cash. Thus, in the beginning each investor will have a true value of $550,000.

Final true value will be the criterion to determine who wins the simulation. During play, however, market value will give some indication of performance. An investor can improve his cash position, and, therefore, both his market value and true value, merely by collecting dividends each quarter on his stock and interest each quarter on his cash. In addition, his market value can improve if the price of his stock is increasing because of the market action of other investors, even though he does not do any trading himself. The commission of $1.20 per share for either buying or selling tends to discourage indiscriminate trading. On the other hand, if the true value of a stock is changing, it will usually pay to actively engage in trading.

The simulation can be thought of as consisting of two separate parts. One of these is the determination of the true value of the stock, as expressed by its earnings. This determination is made entirely by the computer in random fashion. The other is the market price of the stock as determined by the demand for it in the open market. Each of the parts will be analyzed separately.

The True Value of the Stock

In the beginning each share of stock has a market value of $100 and earnings of $1 per quarter on the average. All earnings are paid out as dividends by all companies so that earnings and dividends are synonymous. Actual dividends will not equal the true average earnings in any one quarter but will be generated as a random value from a normal distribution with mean equal to the true average earnings and standard deviation equal to $.06. Thus, although the dividends will sometimes be less and sometimes more than $1, they will, in the long run, average out to $1 if the true average earnings do not change. Interest earned on cash is exactly 1% per quarter so that, on the average, cash is just as good as stock in the beginning.

The true value of the stock will at all times be the true average earnings of the stock times 100. Thus, the true value of the stock in the beginning will be $1 x 100 = $100 or equal to the market value.

The true average earnings (and thus the true value) of the stocks do change, however, as determined by a random generator that sets the probability of a change occurring at .200. When a change in the true average earnings does occur, it can be either upwards or downwards, but it is more likely to be upwards. The probability of it being upwards is calculated as equal to (true average earnings - .40). Thus, in the beginning the probability is .6. If later true average earnings were $1.10, the probability of the next change being upwards would be 0.7. The maximum probability of a change being upwards is 0.8, regardless of true average earnings.

When a change does occur (in either direction) it is equally likely to be either a jump or a trend. If a jump, the size of the change will be a random value from a rectangular distribution ranging from $.05 to $.15 with a mean value of $.10. This will then become the true average earnings until another change occurs and this next change will be a change from this level. If the change is a trend, the size of the trend will be $.02 per quarter, and this trend will continue in the same direction until another change occurs.

The Market Price of the Stock

Each quarter the computer will print out the current price and the dividends of each stock for the quarter, as well as showing the current market value for each investor. Following this, each investor can, in turn, pass or buy or sell exactly 100 shares of one stock, subject to his having the cash if he wishes to buy or owning the stock if he wishes to sell. (A miscalculation here causes him to lose his turn.) When an investor buys 100 shares of a stock, he pays the market price plus a $120 commission. The market price of this stock is then increased by $1 per share. When an investor sells 100 shares of stock, he receives the market price less a $120 commission. The market price of this stock is then decreased by $1 per share. Trading continues until all trading ceases. Passing in any one round does not prevent an investor from trading in future rounds, but any round in which all investors pass is the last round of the quarter and a new quarterly report is issued.

Final Results

True value will determine the winner of the simulation. While market value will tend to give an indication of performance, it can be misleading if some stocks have been bid up (or down) to a price considerably different from the true value of the stock.

At the end, each investor's true value will be calculated as his number of shares of stock times the true value of each share plus his cash.

After each quarter the computer calculates and writes out dividend information on each stock as well as the current price. This forms the primary source of information to the participants concerning what market actions would be appropriate. In addition, the computer writes out the current status of each team, showing the holdings of stock and cash as well as the market value.

Following a brief period to digest the information, the market activity begins with a different team each quarter and goes through the teams in order until all buying and selling is completed. After each purchase or sale the new price is written out. Once the round has begun, fairly quick response is required and a team is assumed to have passed if they are not prepared with a market action.

Although the amount of buying and selling varies with different groups and with different data generated by the computer, a typical play of 15 quarters requires about two hours from start to finish.

Student reactions to this simulation have been very good, with comments typically ranging from "very enthusiastic" to "generally good" and with almost no comments indicating a negative reaction. Many students expressed a desire to play again, recognizing that their original play left quite a little to be desired as compared with any concept of optimal strategy.

In truth, the typical undergraduate student does not make good decisions in the first attempt at this simulation. However, the typical faculty member, department chairman or dean also does not make good decisions in the first attempt. The reasons for this are fairly complex but some factors are noticeable. Even though the rules are distributed well in advance, and would appear to lend themselves to fairly quick analysis (assuming some reasonable level of preparedness), many people either are not willing to take the time to analyze or feel that they can cope with decision making on an ad hoc basis without establishing any general decision rules prior to participating.

Regardless of how much preparation is done, most people will make at least one decision which turns out to have a poor result. The decision may have been a reasonable one, such as paying $105 for a stock that showed a first quarter dividend of $1.14. If the large dividend in truth represented only noise (true value was $100) and the stock changes downward to a true value of $90 in the second quarter and a dividend of $.80, the decision to buy at $105, though reasonable, is quickly shown to have had a poor result. Those who have a general plan for decision making in mind appear far better prepared to cope with this sort of contingency than those playing by the seat of their pants. The decision rule players tend to rationalize that at least their decision was a good one, that you lose some and you win some, and continue to follow their preplanned strategy. Football coaches say that their team won because they stuck to their game strategy even though the opponents ran the opening kickoff back for a touchdown.

However, those playing by the seat of their pants tend to be more emotional about their decisions, and react in several different ways. One of these is to conclude that all dividend information is pure noise and misleading and from then on buy only price. This is a demonstrably inferior strategy and those that follow it can usually be convinced of this after their first experience with the simulation because they end up with a poor result. Another reaction is to feel something like "I paid $105 for it, it is now selling for $96. If I don't buy it now, I'll be admitting my previous decision was wrong; therefore, I must buy at any price below $105." While this philosophy is not usually stated in as definitive a fashion, it is not uncommon to see participants take a position on a particular stock that has them putting all or most of their assets in that stock for no particularly good reason. The interesting thing about this strategy is that while it typically gives very poor results, in some cases the stock selected will take off in later quarters and turn out to be the best of the stocks in true value at the end. When this happens the person taking an end position in the stock in early quarters has purchased it at relatively low prices and becomes a runaway winner of the simulation.

Whether this is desirable or not depends on what the instructor plans to do with the simulation. Either repeated play of the simulation or analysis of the results secured and the reasons for them can usually make it possible to convince the winner of the simulation by this method that he was lucky rather than clever, and that on the average his results will be quite poor. This can provide an interesting takeoff for a discussion of the difference between good decisions and good results, and using the excellent results occasionally secured by this end position strategy to debate a proposition such as "Exceptionally good results necessarily stem from poor decisions" will often provide the student with a far better understanding of the difference between decisions and results than can be secured through more traditional methods.

The interest generated by participating in the simulation makes it a convenient vehicle for introducing many different concepts. In a general policy course, an appropriate discussion would concern the relative merits of poor management of resources, good management of resources, and optimal management of resources. As in real life the search for an optimal strategy has many built-in problems, and though the rules for the simulation are quite precise and relatively few, an optimum strategy is quite difficult to determine. Most people will accept, however, that looking for an optimum strategy at least leads to a good strategy, and even though it is difficult to say how good an optimum strategy would be if it were found, the difference between a good strategy and an optimum strategy is far less than the difference between a poor strategy and a good strategy. The general conclusion that in many problems searching for an optimum solution serves to lead us to a good solution to the problem, and that we make considerable progress thereby, is effectively illustrated.

In a statistics course, one can ask about the expected value of any stock by the end of the simulation. Here it quickly becomes evident that we must know how many quarters will be played. If we are willing to work with expected value for a simplified answer, we can conclude that all stocks will have exactly three changes, and that each change will be for exactly $10. A tree diagram approach can then give an expected value of approximately $107 after 15 quarters. We can realistically ask whether we should not be more detailed and work with the probability of 0, 1, 2, 3, 4, 5, etc., changes and incorporate these into the analysis. The difference between a trend and a jump and the rectangular distribution of size of jump raise similar questions. Students who have had computer programming and some experience in Monte Carlo simulation can write a program to see what happens to many thousands of stocks after 15 quarters, and try to check a theoretical answer in this way. Similarly we could try to find the value of a stock after many quarters (say 10,000) either by analysis or simulation or both.

After we have determined that we are going to play for 15 quarters and satisfied ourselves about what an average stock will be worth after 15 quarters, we still are only making a beginning on the strategy question. The question of liquidity that arises is appropriate to discussion in a finance course. If we believe that an average stock will be worth $107 at the end of the simulation and an apparently average stock is selling for $102, should we spend all of our cash buying the stock? The problem here is that after we spend all of our cash, if a particular stock obviously takes off on a major improvement, we will have to waste a round (as well as paying the commission) selling another stock before we can buy, and the price will undoubtedly be considerably higher because others who had cash will buy during the round, forcing the price up. It is not easy to come up with a good answer to the liquidity question but at least the question is raised in a setting that can readily be understood.

These discussion points all necessarily follow the central question of deciding what techniques are most appropriate in deciding whether or not a stock has changed in value, or, put another way, what the best estimate of the current true value of a stock is. Advocates of sigma decision rules, run analysis, exponential smoothing, and any combination of these and other such techniques can, with some merit, claim their favored technique is best. In truth, if there were not room for a difference of opinion here, the simulation would soon develop into one of most people following similar strategies and therefore one of very little trading. This has not happened with groups I have worked with.

While the analysis and possible takeoff discussions are an argument for simulations of this sort rather than an argument for using a time sharing computer, the interest in the simulation is central to the discussions that arise, and this interest is in large measure due to the fact that the data are being generated on the spot, that the simulation is fairly quick-paced, and that the instructor has no more control over the data or ability to know what is happening than anyone else. I believe that many projects are inhibited by the student's belief that the instructor already knows the answers and he reacts by either resisting or accepting without question. The lack of special knowledge (as distinguished from general knowledge of statistical inference) on the part of the instructor is a major contributor to student interest.

Difficulties with the simulation are mostly those of time and money. One run of the simulation requires about two hours for 15 quarters of play, and buying time at commercial rates is expensive. Analysis and replay can use up a very considerable amount of time and money in achieving the sort of results talked about here, and I have been working primarily with experimental groups rather than using the computer in this way for large classes. Similar programs illustrating other statistical properties are designed primarily for individual use of the computer, and are in effect a form of computer assisted instruction. Even putting a class of 20 students on a terminal for two hours each at commercial rates can tax the financial capabilities of a small private college, and some form of in-house time-sharing facility is almost a necessity. One problem I have been trying to find an answer to is that of projecting the computer results for all to see as they come off the terminal. In the stock market simulation we typically write the stock data on a board and circulate the company data through the room after each quarter. Other programs have more data and make this kind of system unworkable. The best I have done to let everyone in a class see the results coming off the terminal has been to use closed circuit TV with a camera fixed on the terminal, but this has been less than satisfactory with the equipment used. Any suggestions anyone might have for projecting terminal output to a class as it is coming out would certainly be appreciated.

THE COMPUTER IN ELEMENTARY ACCOUNTING*

D. M. Ferguson
University of Minnesota-Duluth
Duluth, Minnesota

Introduction

For many years the student in a beginning accounting course has struggled with columns of figures and sheaves of working papers. The thought that this required arithmetic and writing were inhibiting the learning of accounting occurred to many accounting instructors, and a few institutions did manage to move away from the more procedure oriented elementary course. Practically all accounting teachers, however, agree that a basic understanding of the collection and recording procedures and the organization of accounting information is essential for all students studying the subject of accounting. The primary objective of this project on the use of the computer in elementary accounting is to develop ways to use the computer in the actual instructional process. This integration is intended to relieve students of the drudgery of hand processing, but at the same time require more, and not less, analytical work by the student. A secondary goal of the work is to provide support for the instructor in his task of monitoring student progress and evaluating student work. In order to satisfy the above objectives it has been necessary to develop and test both a set of interrelated programs and a series of special accounting problems. At present, this computerized system is being evaluated and revised based upon a series of tests and experiments which started in the fall quarter, 1970, at the University of Minnesota-Duluth and neighboring institutions.

This paper will examine the needs of the student and teacher which provided the design criteria for the project. Then actual examples of problems will be discussed as a part of the description of the current system of problems and programs. Finally, comments will be offered on the testing of the developed materials and the reactions of students to this change in their learning environment.

Development and Design of the Computerized Accounting System

An early consideration but by no means a trivial one was the requirement for entering data into the computer. One of the more cost effective solutions found, considering potential computer services, was to work with prepared decks of punched cards. In general, not only was the punched card a cheaper alternative than keyboard or tape entry, but the need to structure the problem statement to accommodate the use of punched cards and to eliminate the need for the mistake prone single-finger student keypuncher also contributed to the development of relatively unique problem structures, using punch cards to convey problem information and collect problem work from the students.

For persons studying with the computer, the most critical needs were determined to be easy and individual access to the computerized system, a reasonable response time, and meaningful output and error analysis. The instructor wanted a reliable, low maintenance, and flexible teaching package which would also do all of the busy work related to its use. In addition, the instructor's need for information varied much more than that of the student. As part of his preparation for a class an instructor might want very current information on student progress or, before preparing a test, he might wish to know types and fequency of errors committed by the student on the assigned work. The instructor also wanted controls over the system so that he could make mid-course corrections based on interim measurements such as tests and quizzes. These changes might include an additional work load in a topic area that was presenting difficulty to the class, some special problems designed by the instructor to emphasize a particular point, or a revised schedule of work based upon an evaluation of the progress made by the group.

The choice of programming languages and the configuration of the computer system were determined in part by the available computer equipment and the projected needs of the instructor and his students. The batch environment mentality also affected the development of the system throughout the course of the work because of the rigid nature of requirements such as control cards (these were simplified as much as possible for the student). In fact, the accounting system tends to protect individual student runs from the operating system of the computer, much like a mother watching over her brood, by diagnosing student errors and preventing termination of problems or a batch run.

When the system is complete, there will be three distinct modes in which the student will operate. In the first mode, the individual works in a highly structured situation. Problems are set up in advance (sometimes a prepunched card deck is provided) and the student works with relatively simple tasks. However, once a person is familiar with the process of information exchange which is used in a particular section of problems, the conceptual requirements become more difficult for later problem groups in that section. The second mode is conceived as a service operation where the student seeks mechanical assistance from the computer to aid in his solution of problems. Conceptual movement is from topic to topic with the goal of minimizing the amount of data preparation and maximizing the useful information received. In the final mode, the student is freed from system imposed restraints and asked to do the necessary programming and analysis for real life problems. However, assistance is provided to reduce the work for what might be considered clerical functions such as generating random numbers or printing data on a balance sheet format.

The responsibilities of the instructor are threefold. First, he must predict the needs of the student group in the light of past experience and with regard to the stated content of the course. His predictions appear in the form of a sequence of and emphasis on certain types of problems or topics. Then the instructor must monitor the performance of the actual student group and change his game plan to suit the needs of the class. Finally, a measurement of the student's knowledge in the subject area is taken by the standard testing procedure, but this testing can be constructed with reference to information supplied to the teacher about the work of the class on the computer supervised study.

Current State of the System

During the past two years the major effort of the project staff has been directed at developing problem structures for the beginning mode of student contact which involves the initial and more structured contacts with accounting. Substantial work has also been done to provide support for the instructor across the full range of his responsibilites and needs. Additional work is underway with the second and third modes of student interaction with a time horizon of 2-3 years for completion. For the student, the work in the first mode, which is a controlled and organized situation, three different types of problem structures have been developed.

In accounting, the beginning student is asked to identify and organize certain terms in relation to their significance as a part of accounting terminology. This particular conceptual exercise was isolated and broken down into several parts. First the student is required to organize the account names for a simple balance sheet, then for a simple income statement. Next he must differentiate between balance sheet and income statement accounts by selecting only certain accounts from a prepared list. Of course, the list has been jumbled in a random fashion. Within this problem section there are many different problem groups, each group varying conceptually from a preceding one due to the addition of a new rule or type of account. A sample problem from one of the groups of this beginning section and the computer output for that problem is shown in exhibits 1 and 2. This problem 01-F-S requires that balance sheet accounts be sorted and placed in the correct order under category headings. The students receive instructions and a list of the cards (exhibit 1) and if correctly submitted, output as shown in exhibit 2.

A second requirement considered an essential part of elementary accounting is the translation of a verbal statement into a specific procedural activity. In this case the first encounter the student has with such analysis is completely non-numerical. These verbal entry problems consist of transaction statements prepared under very definite sentence structure rules. After reading the statement, the student selects a specific pair of verbal responses from a prepared list. The verbal commands chosen by the student indicate the changes which have occurred in general accounting categories due to the physical transaction described in the problem statements. An example is the statement: Paid cash for a new truck, which is analyzed as an INCrease to the ASseT truck and a DECrease to the ASseT cash. These sentence puzzles are grouped in relation to accounting activity which would be typical for the business described in the problem name; that is, typical sets of transaction statements differ if the business is a gas station or an amusement park or lawyer's office. Problem groups in this section are differentiated by the type and complexity of the verbal responses and the structure of the word group or sentence. A complete example of this type of problem and computer output is shown in exhibits 3 and 4.

```
UNIVERSAL ADVERTISING AGENCY INC.            SAMPLE PROBLEM          01-F-S

        BUILDING                                         40000   01-F-S01

        MORTGAGE PAYABLE                                 20000   01-F-S02

        SALARY EXPENSE                                   25000   01-F-S03

        PROPERTY TAXES PAYABLE                            5000   01-F-S04

        COMMISSIONS RECEIVABLE                           25000   01-F-S05

        RETAINED EARNINGS                                20000   01-F-S06

        ADVERTISING COMMISSION                           70000   01-F-S07

        CASH                                             10000   01-F-S08

        PROPERTY TAX EXPENSE                              8000   01-F-S09

        LAND                                             25000   01-F-S10

        SALARIES PAYABLE                                  5000   01-F-S11

        COMMON STOCK                                     40000   01-F-S12

        ACCOUNTS PAYABLE                                 10000   01-F-S13

**      ASSETS                                                   01-F-S14

**      CURRENT ASSETS                                           01-F-S15

**      LONG TERM ASSETS                                         01-F-S16

**      LIABILITIES                                              01-F-S17

**      OWNER#S EQUITY                                           01-F-S18
```

EXHIBIT 1

```
TEST  DMFERGUSON            ID  000001            PROBLEM  01-F-S               PAGE 1              DATE

                  UNIVERSAL ADVERTISING AGENCY INC.
                  BALANCE SHEET
                  AS OF MONTH/DAY/YEAR

ASSETS

CURRENT ASSETS

      CASH                                     10000
      COMMISSIONS RECEIVABLE                   25000

TOTAL CURRENT ASSETS                           35000

LONG TERM ASSETS

      BUILDING                                 40000
      LAND                                     25000

TOTAL LONG TERM ASSETS                         65000


TOTAL ASSETS                                  100000

LIABILITIES

      ACCOUNTS PAYABLE                         10000
      SALARIES PAYABLE                          5000
      PROPERTY TAXES PAYABLE                    5000
      MORTGAGE PAYABLE                         20000

TOTAL LIABILITIES                              40000

OWNER#S EQUITY

      COMMON STOCK                             40000
      RETAINED EARNINGS                        20000

TOTAL OWNER#S EQUITY                           60000


TOTAL LIABILITIES AND OWNER#S EQUITY          100000
```

EXHIBIT 2

The third problem section follows in some ways the more conventional problem pattern of elementary accounting texts. Students must learn to prepare numerically defined journal entries after analysis of word statements describing transaction events. Each problem group, however, deals with a specific rule or set of rules and definitions related to an accounting convention or concept, e.g. adjusting entries or closing entries.

Initial problem groups in this section are complementary to verbal entry exercises, but then move to more complex entries and types of businesses not considered previously, e.g. merchandising businesses.

EXHIBIT 3

PROBLEM NAME BAXTER FIX IT SHOP

PAID ONE MONTH'S SALARIES WITH CASH
RECEIVED REPAIR REVENUE ON ACCOUNT FOR FIRST HALF OF MONTH
PAID UTILITIES BILL WITH CASH
RECEIVED CASH REPAIR FEES
PAID OFFICE HELP FOR THE MONTH WITH CASH
PAID FOR REPAIRS TO OFFICE EQUIPMENT GIVING A SHORT TERM NOTE
RECEIVED CASH ON ACCOUNT FROM CUSTOMERS
PAID SHORT TERM NOTE IN FULL WITH CASH
PAID FOR MINOR REPAIRS TO EQUIPMENT
RECORDED USE OF REPAIR SUPPLIES
PAID MISCELLANEOUS EXPENSES WITH CASH
RECORDED REPAIR FEES RECEIVABLE

INC AST	DEC AST
DEC LIA	DEC AST
INC AST	INC REV
INC EXP	DEC AST
INC EXP	DEC AST
INC EXP	INC LIA
INC EXP	DEC AST
INC EXP	DEC AST
INC AST	INC REV
INC EXP	DEC AST
INC AST	INC REV
INC EXP	DEC AST

EXHIBIT 4

```
TEST BCARLSON     ID     000002     PROBLEM 03-B01     PAGE 1

     VERBAL ENTRY STATEMENT ANALYSIS

     PROBLEM NAME   BAXTER FIX IT SHOP

STATEMENT              PAID ONE MONTHS SALARIES
ANALYSIS               INC EXP DEC AST

STATEMENT              RECEIVED REPAIR REVENUE ON ACCOUNT FOR FIRST HALF OF MONTH
ANALYSIS               INC AST INC REV

STATEMENT              PAID UTILITIES BILL WITH CASH
ANALYSIS               INC EXP DEC AST

STATEMENT              RECEIVED CASH REPAIR FEES
ANALYSIS               INC AST INC REV

STATEMENT              PAID OFFICE HELP FOR THE MONTH WITH CASH
ANALYSIS               INC EXP DEC AST

STATEMENT              PAID FOR REPAIRS TO OFFICE EQUIPMENT GIVING A SHORT TERM NOTE
ANALYSIS               INC EXP INC LIA

STATEMENT              RECEIVED CASH ON ACCOUNT FROM CUSTOMERS
ANALYSIS               INC AST DEC AST

STATEMENT              PAID SHORT TERM NOTE IN FULL WITH CASH
ANALYSIS               DEC LIA DEC AST

STATEMENT              PAID FOR MINOR REPAIRS TO EQUIPMENT
ANALYSIS               INC EXP DEC AST

STATEMENT              RECORDED USE OF REPAIR SUPPLIES
ANALYSIS               INC EXP DEC AST

STATEMENT              PAID MISCELLANEOUS EXPENSES WITH CASH
ANALYSIS               INC EXP DEC AST

STATEMENT              RECORDED REPAIR FEES RECEIVABLE
ANALYSIS               INC AST INC REV

     **NO ERRORS**
```

Sample problem 02-D is documented in exhibits 5 through 8. Exhibit 5 is the problem statement and exhibit 6 the list of possible entries the student can make. The entries shown in exhibit 6 refer to an account code which is familiar to the student. Each line on the page represents one card and one journal entry and should be read with reference to accounting terminology as debit account 429 for 18 dollars and credit account 123 for 18 dollars (card 01-D-S01). For every correct entry included in the randomly organized problem statement, the oppostite entry is also included. Thus, the student must really decide between debiting and crediting an account. The computer output for this problem is shown in exhibits 7 and 8 and both the account name and number are printed. Page one lists the cards in the order submitted by the student, but uses a more traditional accounting format to display the information. On page two both the information provided by the student and some data supplied by the pre-prepared solution is printed as an adjusted trial balance.

Within every problem group in all the sections there are 10 problems. Each of these problems is constructed in an identical fashion based on a set of rules for construction of the problem, but the type of business selected for the problem which helps define a normal set of transactions is usually not repeated within a group. Twenty-five different types of firms compose a comlete list of businesses which appear in the prepared problems of all sections. One reason for the additional problems is to allow the instructor to vary the number of problems assigned on a specific conceptual topic in light of the emphasis he feels necessary. A second reason for the expanded problem group is to allow random assignment of problems within a group. In this way, the instructor knows that over a number of problem groups each student has an individualized assignment. Finally, a student may feel he needs additional work with a certain type of problem and the existence of additional problems within a group gives him the opportunity for further practice.

EXHIBIT 5

Section 02
Group D
Sample Problem

Doctor Chester Cap, D.D.S.

Dr. Cap, D.D.S.
Trial Balance
April 30, 1945

Cash	$ 781.00	
Prepaid Insurance	216.00	
Office Supplies	185.00	
Dental Equipment	3,456.00	
Accumulated Depreciation, Dental Equipment		$ 72.00
Accounts Payable		450.00
Chester Cap, Capital		4,000.00
Professional Fees		736.00
Salary Expense	250.00	
Telephone Expense	15.00	
Utilities Expense	5.00	
Rent Expense	350.00	
	$5,258.00	$5,258.00

Other Data

Dr. Cap has not paid his secretary for the second half of the month. She has earned $250.

On the 4th of this month Cap purchased a one year insurance policy for $216.

Office supplies worth $35 were used during the month.

When Dr. Cap started his practice in January, he purchased dental equipment for $3,456. He estimates it will have a useful life of 12 years.

At the end of the month he had completed work for patients in the amount of $325, which has not been recorded.

INSTRUCTIONS: Make the necessary journal entries to adjust the books at the end of the month.

```
DR. CHESTER CAP                                 SAMPLE PROBLEM       02-D

429      18 /123       18                                      02-D-S01

446     250 /211      250                                      02-D-S02

181      24 /438       24                                      02-D-S03

408      24 /181       24                                      02-D-S04

118      35 /439       35                                      02-D-S05

123      18 /429       18                                      02-D-S06

126     235 /313      235                                      02-D-S07

211     250 /446      250                                      02-D-S08

439      35 /118       35                                      02-D-S09

313     235 /126      235                                      02-D-S10
```

EXHIBIT 6

```
TEST  DMFERGUSON               ID 000001           PROBLEM 02-D          PAGE  1         DATE

      DR. CHESTER CAP
      GENERAL JOURNAL

      ACCOUNT NAME                                          NUMBER        DEBIT      CREDIT
ENTRY

  1
      INSURANCE EXPENSE                                      429           18
           PREPAID INSURANCE                                 123                       18
  2
      SALARY EXPENSE                                         446          250
           SALARIES PAYABLE                                  211                      250
  3
      DEPRECIATION DENTAL EQUIPMENT                          408           24
           ACCUMULATED DEPRECIATION DENTAL EQUIPMENT         181                       24
  4
      PROFESSIONAL FEES RECEIVABLE                           126          235
           PROFESSIONAL FEES                                 313                      235
  5
      OFFICE SUPPLIES EXPENSE                                439           35
           OFFICE SUPPLIES                                   118                       35
```

EXHIBIT 7

```
TEST  DMFERGUSON                   ID  000001              PROBLEM  02-D              PAGE   2

                 DR. CHESTER CAP
                 ADJUSTED TRIAL BALANCE

      ACCOUNT NAME                                             NUMBER        DEBIT     CREDIT

      CASH                                                      102           781
      OFFICE SUPPLIES                                           118           150
      PREPAID INSURANCE                                         123           198
      PROFESSIONAL FEES RECEIVABLE                              126           235
      DENTAL EQUIPMENT                                          142          3456
      ACCUMULATED DEPRECIATION DENTAL EQUIPMENT                 181                       96
      ACCOUNTS PAYABLE                                          201                      450
      SALARIES PAYABLE                                          211                      250
      CAPITAL ACCOUNT CHESTER                                   252                     4000
      PROFESSIONAL FEES                                         313                      971
      DEPRECIATION DENTAL EQUIPMENT                             408            24
      INSURANCE EXPENSE                                         429            18
      OFFICE SUPPLIES EXPENSE                                   439            35
      RENT EXPENSE                                              444           350
      SALARY EXPENSE                                            446           500
      TELEPHONE EXPENSE                                         451            15
      UTILITIES EXPENSES                                        454             5

                                                       TOTAL                 5767       5767

      **NO ERRORS**
```

EXHIBIT 8

For the type-one problem, data preparation is handled exclusively through the manipulation of pre-prepared card decks. Problem decks often contain bogus cards and are always randomly organized. When a student wishes to work a problem he requests a card deck, performs the required sorting, and submits his problem to the computer. Verbal entry problems always use prepared statements but some groups require the student to prepare the analysis cards, which define the verbal reponse in terms of accounting procedures appropriate for the transaction statement. Journal entry problems allow use of the prepared card deck format, student preparation of journal cards, or experimental formats where increases and decreases can be used in place of debits and credits.

The problem output and the accompanying Error Analysis is without question the most important feature of the system as far as the student is concerned. Computer output is a standard feature of all problem submissions; however, there is no guarantee that the formatted accounting output will be correct. If the student submits input data which is incorrect he looks to the error analysis section to assist him in understanding his mistakes. Error messages differ by sections but in type one problems messages are keyed to account names and categories, in verbal entries the error notation appears after each entry, and in the journal entry problems it is keyed to final account balances. Samples of error messages are shown in exhibit 9 and these are generated by the computer when it compares the pre-prepared solution with the student data.

EXHIBIT 9

Error Messages

Section One

1. This category heading is out of order.
2. A category heading was omitted.
3. An account was omitted from this category.
4. An improper account was included in this category.
5. An account was omitted from the problem.
6. There are too many categories for this type of problem.
7. There are too many accounts in a category.

Section Two

1. Trial balance does not balance.
2. This account does not have the same balance as in the solution.
3. This account is missing from the student ledger.
4. This account does not appear in the solution.
5. An asset or expense account has a credit balance.
6. A liability, revenue, or owner's equity account has a debit balance.
7. Expense or revenue accounts not closed when required.
8. Accounts were closed before all adjustments were made.

Section Three

1. This entry in the above analysis is incorrect.
2. The above transaction cannot be identified.
3. There are too many transaction or analysis cards.

The instructor's initial contact with the system is the preparation of the term's study assignment. A program which selects problems randomly from groups in the sequence selected by the instructor and which prepares assignment sheets for distribution to the class is used first. Several utility programs which set up data and program files are also run in order to prepare for the quarter's work. After some data has been collected by the system, the instructor can begin to exercise his other options.

First, there are status reports which provide information on the problems attempted by a student and the number and types of errors committed on those problems. Problem analysis is the second type of report which is prepared for the teacher and it is keyed to the errors committed by the students. Other measurement information, such as tests or grade point averages are also used in conjunction with the error information to perform investigations similar to item analysis of multiple choice exams but are keyed to performance on problems from selected groups. This information can be put into use when preparing tests, when altering content coverage to meet unanticipated needs, or when making the next term's forecast of course content. These various analyses are facilitated by the basic design of the problem file which is divided along conceptual and procedural dimensions which assist in the isolation of various conceptual difficulties. The final instructor contact occurs when the instructor requests a weighted ranking of his students based upon the various input data supplied by the instructor and the student performance on the system generated assignment.

Technical and Statistical Information

Over 50 programs and 300 problems are now a part of the system. Both FORTRAN and COBOL have been used, but the greatest part of the programming has been done in FORTRAN. Although tape files are used as backup, the program and data storage is on disk files and presently requires about 6 million characters in order to service 250 students. When operating, the programs work in much the same manner as the computer's own master system. All accounting and process control is handled internal to the system programs. Execution time is a function of the size and type of problem, but averages 6-10 seconds for type 1 problems and 15-30 seconds for journal entry problems on a CDC 3200. Both programs and solution data necessary for each student's problem are located on the disk and brought into core before processing the problem.

Results of Testing and Observations

Since the student is able to approach the computer as often as necessary and primarily at his own convenience, the system has the potential for significant impact on the learning process. The student can even decide for himself to do more work in areas identified as his conceptually weak spots through computer error analysis and he will not tire his computer instructor with insistent questions--and the computer is not upset if the student waits until the final week to hand in all his problems. Although batch processing is verboten in some circles, the entire system of programs and problems has been designed to function effectively in a batch environment. Preliminary experimentation with as few as two turn-arounds per day indicate that this response time is adequate to sustain the student's interest in learning accounting through the computer. Finally, both the novelty and the help a student receives seem in general to cause more students to develop an interest in learning accounting.

The instructor in turn has a much more refined tool for influencing the behavior of his students in those directions necessary for learning accounting. He knows sooner and in a much more detailed manner the problems his entire class is having. When students falter in a given area the instructor can respond by assigning more problems in certain groups (or modifying class work) which will give the students more practice in trouble areas and hopefully reinforce correct concepts and procedures.

Even the fact that each student has an individualized assignment, because problems are selected on a random basis from within a group, helps both the student and the instructor. The student has a different assignment than his roommate or friend has but is working on the same kind of transactions. Students can still discuss their conceptual difficulties, but really won't just borrow the answers. An instructor could rely on this random assignment to force all of his students to do some work, even if they actively seek outside support, since it will be difficult to locate other students with anything close to exactly the same set of problems assigned.

Students in the test section have also made some interesting points. First, they can perform the anlysis needed to initially execute a workbook problem in 1/3 to 1/2 of the time required for comparable hand written problems. Second, certain texts devote so much space to presenting the hand method that it sometimes is difficult to get the information out of the text to assist in the analysis orientated workbook problems.

On the negative side is the change in student expectations which tends to place a larger burden on the staff. When students learn that they can get answers even twice a day, they begin to feel that it is their right to get those answers or that service at least that often. Computers, however, can have multitudes of hardware and software problems, not all at convenient times. Taking all this in stride, both signs and error memos are used to communicate to the student the current status of operations (Everything OK, Computer Delay, Program Error, etc.). In addition, when errors are discovered in the problem file (usually by students) a memo is passed out so other students may be made aware of it.

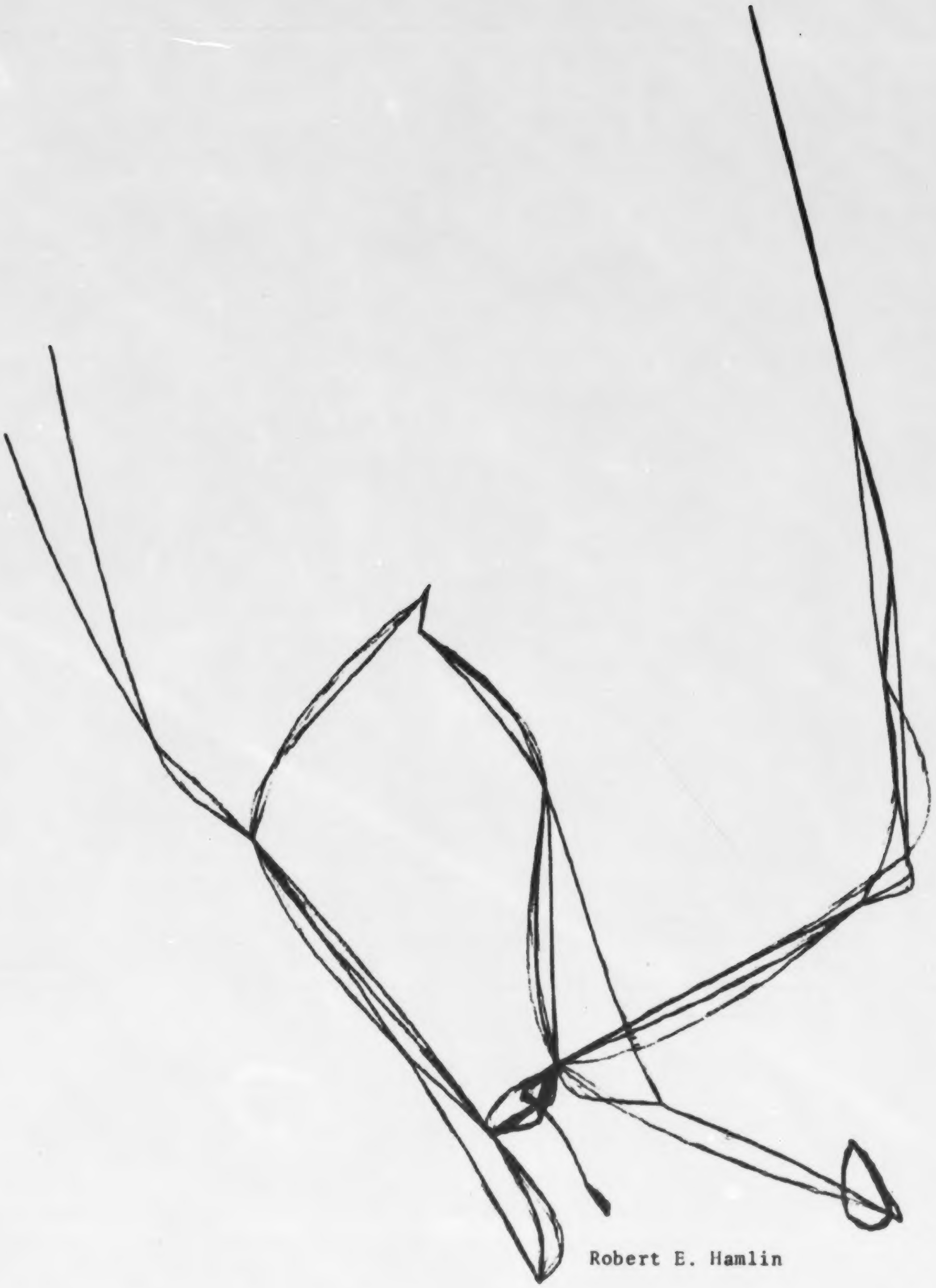
Final Comment

The potential improvements of computer supported instruction are myriad and exciting--and the development costs staggering. In these times of financial and external pressures, the small college can best participate in the development of computer materials on a cooperative basis or through exchange of experiences in conferences such as those sponsored at Iowa in 1970 and Dartmouth in 1971.

Whether the computer is used as a novel supplement to or an integral part of the learning process, it seems clear that many more applications of the computer to undergraduate instruction will be tested in the next few years. In particular, elementary accounting instruction may become much involved with the computer when materials such as those described in this paper are available for general use.

ACKNOWLEDGMENTS

* This paper was made possible in part by funds granted by the Center for Curriculum Studies, University of Minnesota, and the 3M Company, St. Paul, Minnesota.



Robert E. Hamlin

COMPUTER SIMULATION OF CHROMATOGRAPHIC COLUMNS

Stephen Prager
University of Minnesota
Minneapolis, Minnesota 55455

During the past several months we have been using the Culler-Fried (CF) system at Santa Barbara in a course on separation processes. A number of programs have been written in this connection, illustrating various aspects of distillation, zone refining, and chromatography. This paper describes the third group, which has been more extensively developed and may, in view of the importance of chromatographic methods in analytical chemistry, prove to be generally useful.

The special features of the CF language are (1) that it can work with lists of numbers (vectors), with each operation being performed on the entire list, and (2) that two such lists may be stored in an abscissa and an ordinate register respectively and plotted against one another on an oscilloscope display. This arrangement lends itself very naturally to the generation of solute distributions in chromatographic separations.

Our programs are based on the well known analogy between chromatography and the Craig multiple extraction scheme illustrated in Figure 1. In the Craig arrangement the mixture to be separated is dissolved in a solvent β, loaded into tube (a) and treated with an extractant α in tube (1). The residue in tube (a) is then extracted further with fresh in (2), and the contents of tube (1) equilibrated with a sample fresh β in (b). As this process is repeated, the α and β phases move in opposite directions, α to the right and β to the left. Each solute in the mixture will, on the average, move to the right or left according to the degree to which it favors either the α or the β phase.

If after equilibration a fraction p of a given solute is in the α phase, then, at any stage of the Craig process, a molecule of that solute will move to the right with probability p, and to the left with probability (1-p). This uncertainty reflects itself in a gradual broadening of an initially sharp solute peak, a broadening which is superimposed on the motion of the peak itself. After n steps (i.e. n/2 displacements of phase α to the right alternating with n/2 displacements of phase β to the left) the solute peak will have moved $n(p-(1-p))$ steps to the right and attained a width extending over $\sqrt{2np(1-p)}$ tubes.

A chromatographic column is essentially a continuous version of the Craig scheme, and can be represented by a Craig apparatus consisting of a very large number of tubes. We use two vectors α and β of 100 elements, each element corresponding to a tube. The α vector represents the solute distribution in the α phase, the β vector the distribution in the β phase. Initially all elements are set equal to zero, except for element 50 of the vector, into which we load the amount of solute being chromatographed. If there are several solutes, as will be the case in practice, a separate pair of α and β vectors must be used for every solute present.

The CF language provides a "right shift" command (RS), which moves every element of a vector one step to the right, and an LS command which similarly moves every element to the left; the operation of a Craig apparatus as in Figure 1 can thus be imitated directly, with a minimum of mathematical manipulation. The basic sequence for the development of the initial solute distribution is shown on the following page.

At the end of the sequence the total solute ($\alpha+\beta$) in each element is plotted against the index of the element and displayed on the oscilloscope screen. Figure 2 shows a series of such displays for the separation of a pair of solutes.

The parameters under the control of the student running the program are the initial amount of each solute and the distribution coefficient p; the program is written for two solutes, so that four entries must be made at the beginning. After the initial data have been supplied, pressing a single key will develop the distributions of both solutes by two steps at a time (i.e. displacement of α phase one step to the right and β phase one step to the left) and display their sum. Auxiliary programs display instructions and permit display of individual solute distributions as well as adjustment of the ordinate scale.

```
LOAD α                      Loads initial distribution

RS                          Shifts α phase one step to right

+ β                         Gives total solute in each
                            element (phases α and β together)

· p        ]                Computes solute in α
STORE α    ]                phase after equilibration

· (1-p)/p  ]                Computes solute in β
STORE β    ]                phase after equilibration

LOAD β     ]
LS         ]
+ α        ]
· p        ]                Repeats previous program,
STORE α    ]                but shifts phase β to left
· (1-p)/p  ]                instead
STORE β    ]
```

Several modified versions of this program have also been prepared to demonstrate various factors influencing peak shapes. If the contact time t between the two phases after each step is not sufficiently long compared to the relaxation time τ required for equilibration, the solute peaks spread more rapidly than expected from the previous analysis. If the ratio t/τ is less than unity, considerable skewing of the peak shape also occurs. Displays of this sort are readily generated (Figure 3) by rewriting the routine which calculates the solute distribution at each step. Thus instead of

$$\ldots\ldots\ (\alpha + \beta) \cdot p \text{ STORE } \alpha \quad \ldots\ldots$$

we put

$$\ldots\ldots\ (\alpha + \beta) \cdot p + (\alpha(1-p) - \beta p) \cdot e^{-t/\tau} \text{ STORE } \alpha \quad \ldots\ldots$$

It should be noted that all the plots in Figure 3 involve only a single solute; the possibility of false multiple peks arising from poor contact between the phases is clearly shown in Figure 3c.

Distortion of peak shapes can also occur in the absence of equilibration difficulties, if the equilibrium relation between the solute contents of the two phases departs from the simple proportionality assumed so far. Thus Figure 4 shows what happens if the β phase saturates with solute according to a Langmuir isotherm:

$$\beta = \left(\frac{1-p}{p}\right)\left(\frac{\alpha}{1+W\alpha}\right) \begin{array}{l} \xrightarrow{\alpha \text{ small}} \dfrac{(1-p)}{p}\,\alpha \\ \xrightarrow{\alpha \text{ large}} \dfrac{(1-p)}{pW} \end{array}$$

In effect this produces a situation where the solute favors the α phase more at high than at low concentrations. The peak of a solute distribution will thus move to the right more rapidly than the wings, leading to a sharpening of the right wing and a broadening of the left.

Finally the effect of an association equilibrium between two solutes 1 and 2 in the phase according to the reaction

$$1 + 2 \rightleftharpoons (12)$$

is shown in Figure 5b. The tendency of the two solutes to move together can be clearly seen from the unequal heights of the two peaks (equal amounts of each solute were loaded onto the column initially; the distribution which would have been obtained in the absence of association is shown in Figure 5a).

All of these programs were written for limited interaction with the student. That is to say, the student had control of the parameters (p, t/τ, W, etc.) governing the operation, but no control over the program itself. We felt this to be necessary at the present stage of our use of the Culler-Fried system; since it appeared unlikely that the same group of students would have a second chance to use CF language in another course, asking them to learn how to write their own programs seemed unjustified. As a result a certain amount of console time was spent rather unproductively by students trying to decode instructions displayed on the oscilloscope screen (we observed several students patiently copying the instructions down word for word, until we made use of our teletypewriter to produce a printed version). More serious was the fact that since we did not tell the students in detail how the programs were constructed, it was very difficult for a student to repair an error short of starting over from the beginning. However, all programs did function in the end with a minimum of supervision on our part, and the two consoles which we had available proved adequate to handle the eight students in the course while still leaving ample time for other users.

While there is no doubt that the most effective use of the system can only be made by students doing their own programming, the benefits of this intermediate level of computer-assisted instruction are by no means negligible. One of the principal difficulties faced by students in the physical sciences is the visualization for various parameter values of relations developed in class. We believe that use of the display facilities provided by the CF system is a very effective way of dealing with this problem.

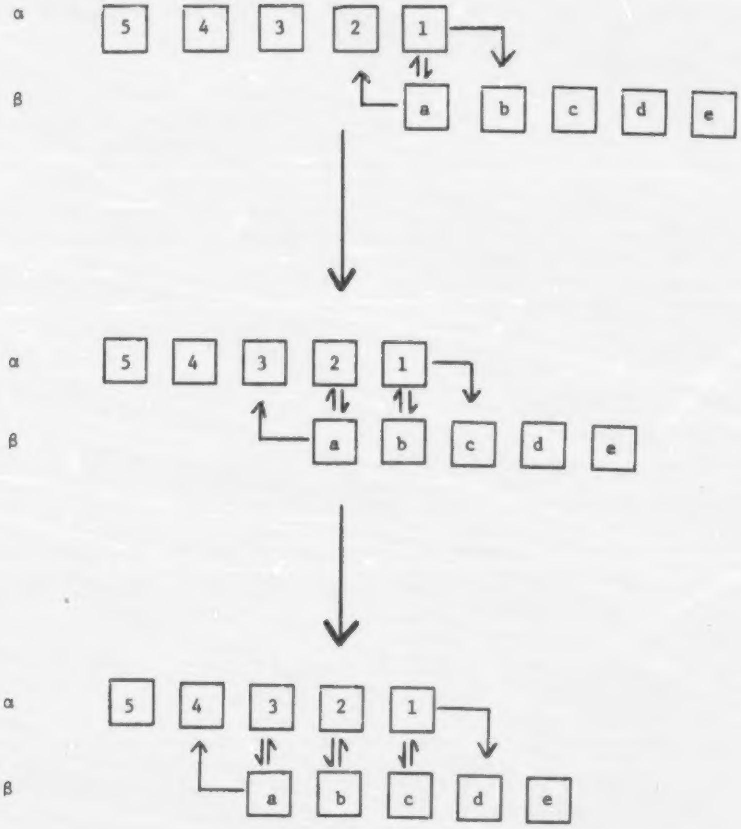

α
5
4
3
2
1
β
a
b
c
d
e
α
5
4
3
2
1
β
a
b
c
d
e
α
5
4
3
2
1
β
a
b
c
d
e

Figure 1

The Craig Multiple Extraction Scheme

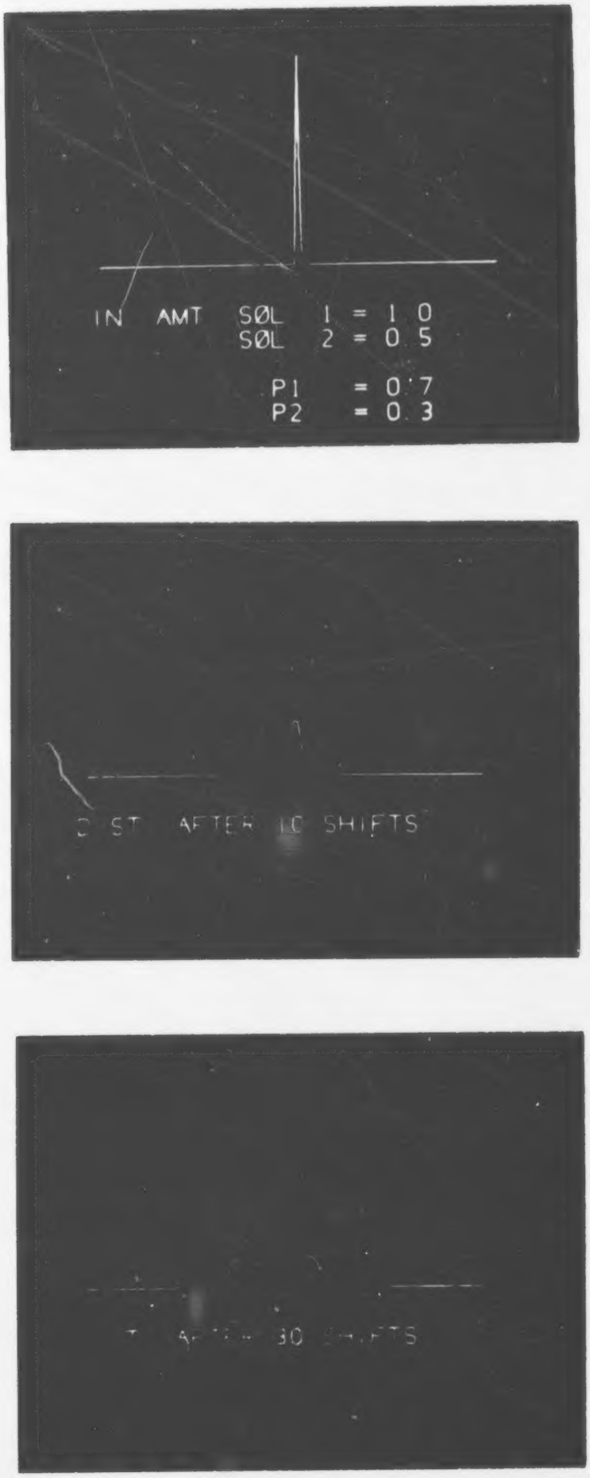

IN AMT SOL 1 = 1.0
SOL 2 = 0.5
P1 = 0.7
P2 = 0.3
DIST AFTER 10 SHIFTS
DIST AFTER 30 SHIFTS

Figure 2. Separation of two solutes

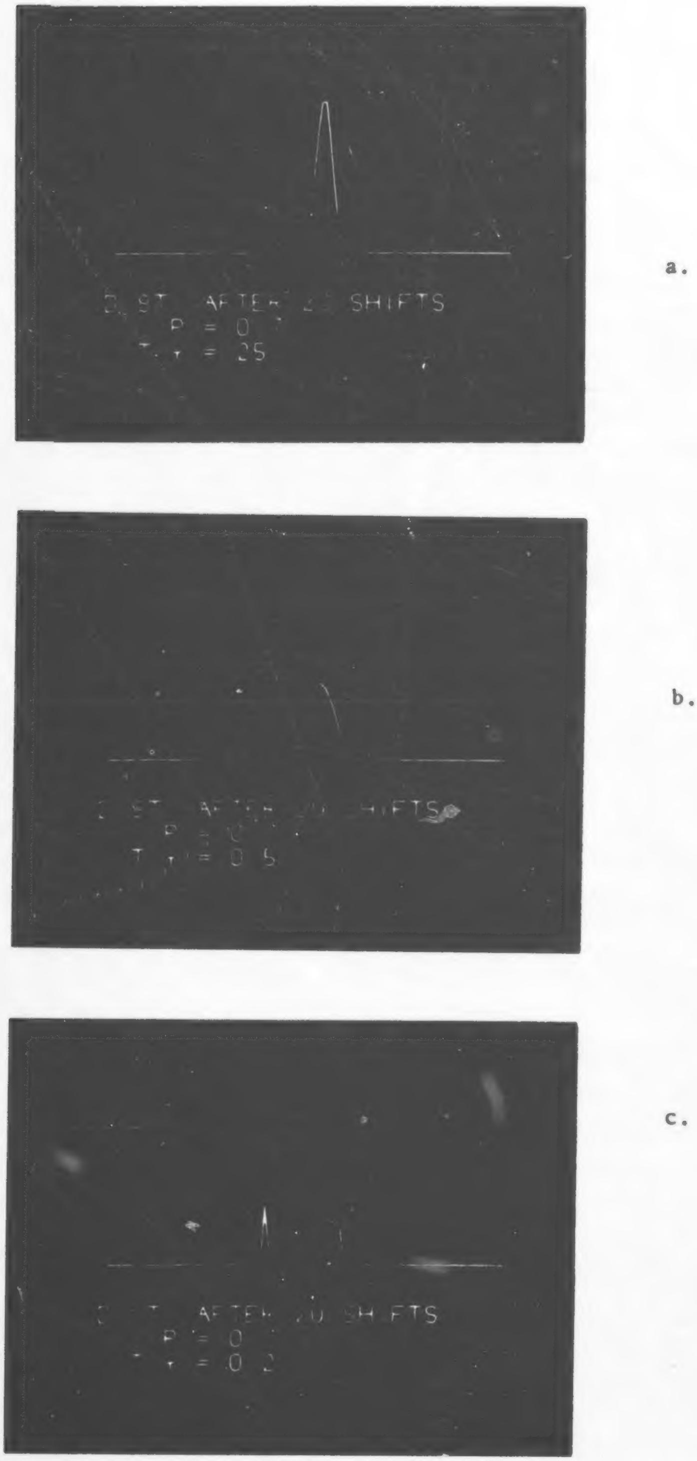


Figure 3. Effect of contact time on distribution of a single solute

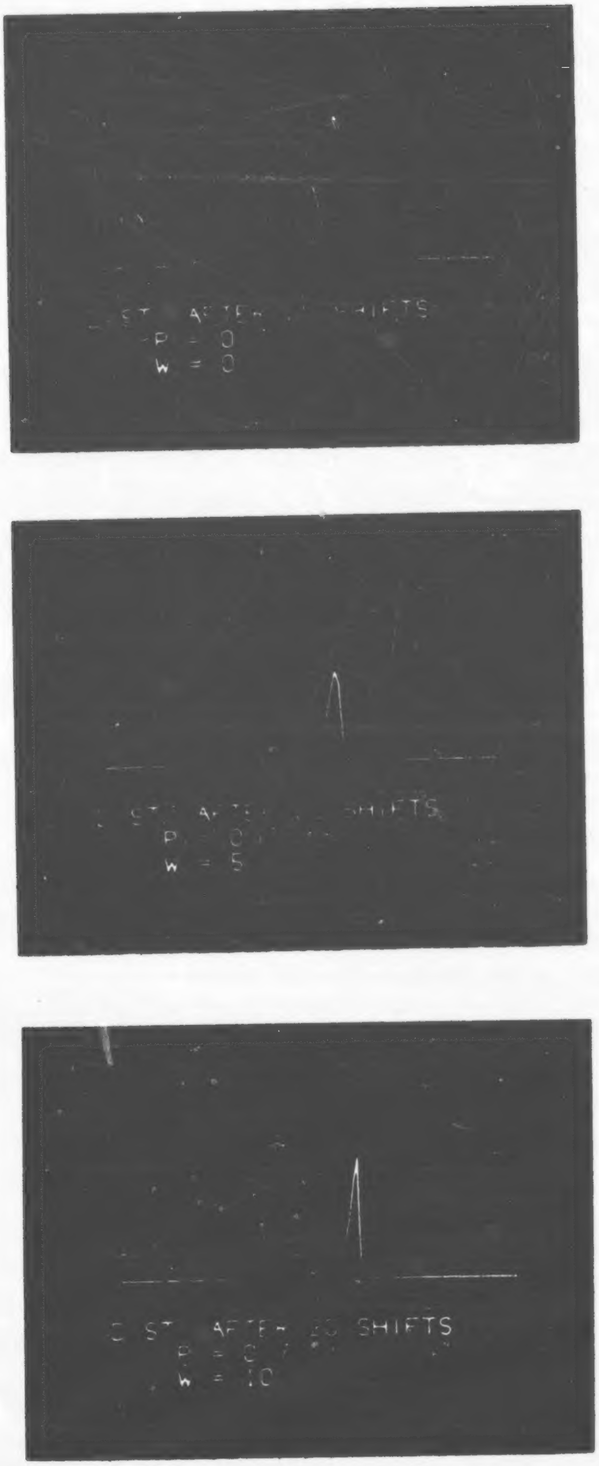


Figure 4. Effect of saturation (single solute)

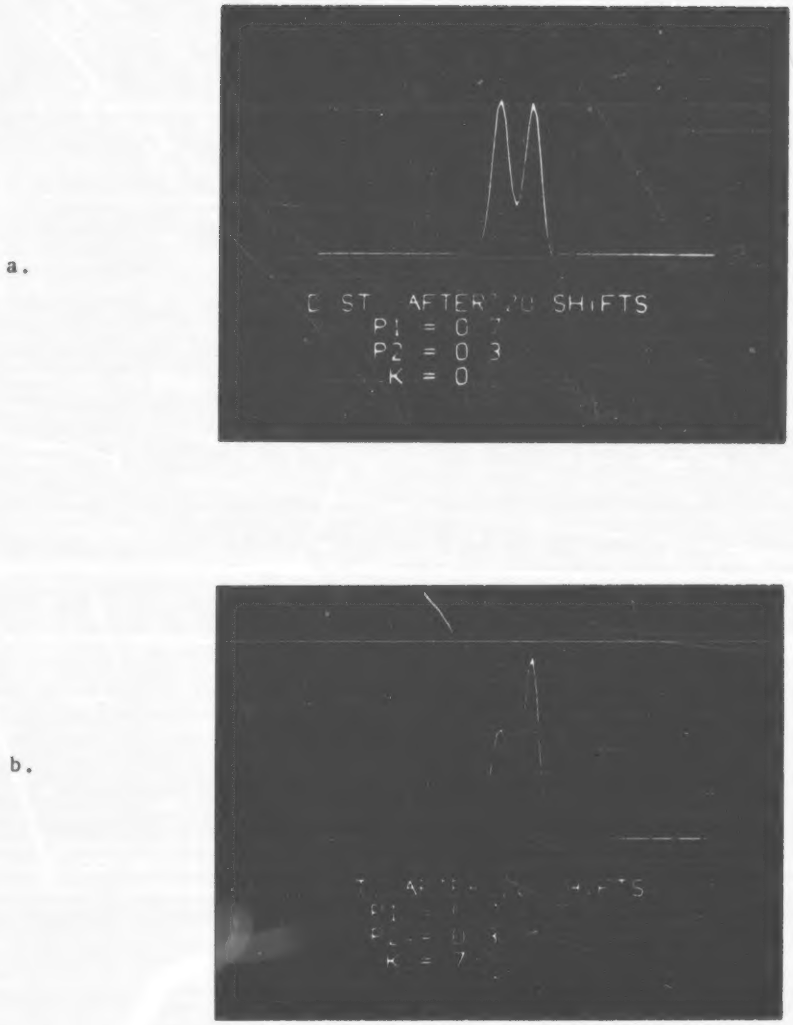


Figure 5. Effect of association on separation of two solutes
(K is association equilibrium constant)

A LIVE INFRARED DATA SYSTEM FOR CHEMISTRY EDUCATION

Joseph R. Denk
Curriculum Development Manager
North Carolina Educational Computing Service
Research Triangle Park, North Carolina 27709

The Dow-ASTM file of infrared spectra for 92,000 compounds has recently been made available for computer searches by several commercial firms. (1) ISIS, Infrared Spectral Information System was developed for accessing this file by the North Carolina Educational Computing Service, NCECS, in cooperation with the Triangle Universities Computation Center, TUCC. (2) As far as we know, ISIS represents the only search system on this "live" data base that has been built for educational usage, both for classroom and research applications.

The Dow-ASTM file of 92,000 compounds contains coded spectral information for 87,000 organic compounds and 5,000 inorganic compounds in the infrared and far-infrared regions, respectively. Each record in the file also contains a large amount of information on the structure and physical properties of the compound. All of the file entries have been abstracted from the literature and ASTM has proposed to update the file at a rate of about 10,000 compounds per year. The sources of the entries of the current file are:

American Petroleum Institute Project 44
Sadtler Catalog of Spectrograms
NRC-NSB File of Spectrograms
Spectrograms Abstracted by ASTM-sponsored Groups
Documentation of Molecular Spectroscopy
Coblentz Society Spectrograms
Manufacturing Chemists Association
Infrared Data Committee of Japan

The file is considered by the author to be live both because of the presence of each entry in the literature (no simulated spectra) and because of the updating potential.

ISIS (2) was built for the IBM 360 model 75 at TUCC. This computer center services 48 institutions of higher education and 10 high schools and offers disk capacity for such a huge file. The original Dow-ASTM data base was converted from card to disk format for more economical searches based on educational needs and to open the entire file to TUCC users; the latter being impossible for more limited configurations. An economical single pass searching system together with several systems advantages available were put together with the data base to bring the cost down below $1 per search.

ISIS will be discussed under three titles, "The Contents of the Data File", "How ISIS Functions for the Chemist", and "Usage of the System Both for Research and Education". Finally, a summary of the availability and transportability of the system will be given together with current developments of the system.

I. The Contents of the Data File

The spectrum for each entry is coded for band location in the infrared region (1.0 to 15.9 microns) or the far-IR region (11.0 to 35.0 microns). Band locations are coded to the nearest 0.1 microns. The only bands recorded are those which have an absorption of above 30 percent based on the strongest band being above 90% absorption. Strong bands are also coded for their appearance in any 1.0 micron range.

Structural and classification information differs for organic (IR range) and inorganic (far-IR) entries. For organic compounds, the presence of certain elements or groups of elements are available (Table 1). General structural information such as position(s) of unsaturation is summarized in Table 2. Tables 3 and 4 show the information available for certain structural groups containing oxygen, nitrogen, or sulfur.

Inorganic classification information is available for elements and radicals (Tables 5 and 6.)

A limited number of both organic and inorganic compounds have melting or boiling point information as well as empirical formula data for carbon, nitrogen, oxygen, and sulfur. Finally, each entry has a serial number corresponding to a catalogue entry or related to a literature reference. This serial number also indicates whether the entry is IR or far-IR and adds a code for special entry meanings (20 publisher-assigned indicators).

II. How ISIS Functions for the Chemist

ISIS allows the approach to this data base through several directions. The most obvious direction is to submit a spectrum and search the file for matching spectra in order to produce identification (organic qualitative analysis). But several other approaches to the data base are discernible. ISIS was built and additions are under development so that the entire contents of the file become available through a combination of five simple one-word commands to the computer. This requires a flexibility to ISIS the working of which is the subject of this section.

The first command necessary to any user of ISIS is a RUNTYPE statement. At present the only documented usage of ISIS is to do a spectral search with an unknown spectra, an option we call SPECTRA. Other options now under development include searching just structure for a spectra, and just empirical formulae for structure or spectra. A RUNTYPE statement looks like:

```
RUNTYPE                    SPECTRA
```

Several options are available for each RUNTYPE. For a spectral search, the following OPTIONS are available:

HITLIM	Limit of the hits in the file. Default = 50
WIGGLE	Wiggles the spectrum fed into the file ± 0.1 microns and searches with the resultant spectrum.
ORGANIC, INORGANIC	Searches can be made on these subfiles.
IR, FARIR	Either subfile may be searched.
PRTPEAKS	Printout of recorded peaks on hits can be made.
PRTCLASS	Chemical classification information can be printed for hits made

An OPTIONS statement can be as follows:

```
OPTIONS         WIGGLE=YES,PRTPEAKS=YES
```

A third command type for spectral searches is the INCLUDE/EXCLUDE statement. With this optional command, chemical classification or structural information can be added to a search. The form of this command can be:

```
INCLUDE   C=O,OC2H5
```

This command is still under development.

The fourth command is the NOBAND statement. Regions of no predicted recorded bands can be entered in the search. The form of the NOBAND command can be:

```
NOBAND (3.5,5.5),(10.0,12.0),(12.1,13.7)
```

This command is essential for efficient spectral searches.

Command 5 is the coding of the spectrum itself. A hypothetical spectrum and its coded peaks are shown in Figure 1. Only bands above 30% absorbance are coded in the original file. This criterion was based on the strongest peak being over 90% absorbance. (An elaboration of the spectral input criteria is treated in the User's Manual.) A sample command for a spectrum can be:

```
SPECTRUM 5.9,6.3,8.3,12.1,13.4,14.6
```

A combination of commands can be seen in Figure 2 illustrating sample input and output for an "unknown" spectrum (which happened to be ethyl benzoate). (3)

ISIS thus allows flexibility in searching the file for any information. The spectral coding, the classification and structural information, and the serial numbers are all available for searches in any desired direction. ISIS also allows combinations of these information categories to be built for any search desirable. The WIGGLE option allows for spectrophotometer differences and standardization problems as well as recording errors. PRTPEAKS helps in isolating answers from a large number of hits due to a low amount of input information. The various RUNTYPES give the searcher a chance to go to the file for any part of the record on each entry or for several entry records corresponding to a certain amount of input information.

III. Usage of ISIS Both for Research and Education

The original usage obviously intended for the data base was for the qualitative identification of unknown spectra from published spectra. This "organic qual" application has been useful both for undergraduate organic "Qual" courses as well as for research. Several colleges in the NCECS network have used ISIS for these purposes. Heavy users are attempting to subset the original file for commonly used "unknowns" in organic "qual" courses so that cost reduction can occur and so that additional information may be added to each record. A subfile of 500 or so compounds may be much more desirable than the entire file. Addition of missing empirical formula data and results of typical qualitative analysis tests, such as the azoxybenzene-aromatic test, can be made on a user defined subfile. Subfile building is also a part of the ISIS system.

Usage in qualitative analysis would not have given sufficient reason to develop ISIS at NCECS. The purpose of developing a flexible system to cover the entire data base was to allow the student or researcher to pursue structural effects on infrared data that was real. It is felt by the author that this function of the computer provides the possibility for introducing creativity in student problems or projects. The influence of structural variation on spectra has not been fully examined and ISIS allows studies that are providing innovative approaches to curriculum. To use an analogy, ISIS is a step toward original work in undergraduate chemistry just as IMPRESS is at Dartmouth for sociology.

Two universities in North Carolina, North Carolina A and T State and Wake Forest, are using ISIS to pursue the effect of structural variation on infrared spectra as part of the development of the system. Students are to do semi-quantitative estimates of such effects as electron feeding or withdrawing on an infrared-active site by studying spectral changes for various compounds substituted with groups thought to produce these effects. It must be noted that the work will be at best semi-quantitative. This limitation is due to the absence of absorption maxima in the file and to the restriction in the spectral coding of band locations to the nearest 0.1 microns. Despite this limitation, macro changes in spectral bands became easily detectible and the coding of strong bands somewhat helps the problem due to the absence of absorbance maxima.

North Carolina A & T State University is also using ISIS as part of an updated chemical literature course. Automated searches have not been possible to date due to cost limitations. ISIS makes the use of this automated file cost-effective for such a course.

Finally ISIS has been made available for research on machine learning at the laboratory of Charles N. Reilley of the University of North Carolina at Chapel Hill. The information on the file is being used to teach the computer to distinguish spectra and to kick-back machine information as its learning process. Hopefully, this will add to chemical information unavailable by any other method.

IV. Current Developments, Availability, and Transportability

NCECS and TUCC are currently working to add to the data base a name-to-serial number file which would allow the user to input or output a chemical name instead of a code number. This name-to-serial number file is available from ASTM both automated and in the form of a $52.00 paperback. The automated facility should be working before the Dartmouth conference.

As was mentioned, the INCLUDE/EXCLUDE option is under development. This option is working at TUCC and only needs refinement. Different search types, RUNTYPE commands, are being developed to search with only structural information, empirical formula information, or serial numbers (names).

ISIS is transportable for IBM 360 installations for a price (to be determined). I/O is written in PL/1 while the search system is in 360 Assembler language. The data base with serial number is available from ASTM for $950. The name-to-serial number file is also available from ASTM for an additional $950. A copyright clearance is necessary for both ISIS (from TUCC) and the data base (ASTM).

Other systems searching the same data base are available. Interested parties should contact Mr. R. Sherwood of ASTM. ASTM itself markets an 1130 version.

ISIS searches can now be made for less than $1.00 to educational users. This makes the classroom usage possible, especially for "organic qual". The development of this low cost system was made possible through a grant and other valuable assistance from the R. J. Reynolds Tobacco Company of Winston-Salem, North Carolina.

FOOTNOTES

1. Among these are Sadtler Corp., Singer Corp., and Dow Chemical Corp.
2. J.R. Denk and J. Gunn, "ISIS, Infrared Spectral Information System--User's Manual, Copyright, Triangle Universities Computation Center, Document No. LSR-98, 1970.
3. The spectrum of ethyl benzoate was reproduced from John R. Dyer, Applications of Absorption Spectroscopy of Organic Compounds, Copyright 1965. By permission of Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

TABLE 1

SUMMARY OF CODING FOR ELEMENTS PRESENT IN ORGANIC COMPOUND ENTRIES

(Each square represents a single code)

O	Fe, Co, Ni, Mn
N	Cr, Mo, W, V
S	V, Nb, Ta, Pa
F	Ti, Zr, Hf, Th
Cl	Sc, Y, La, Ac
Br, I	Ru, Rh, Pd, Os, Ir, Pt
P, Bi	Rare Earths
As, Sb	Li
Si, Ge	Na
Sn, Sb	K
B, Al	Rb, Cs
Se, Te, Po	Be
Ga, In, Tl	Mg
Zn, Cd, Hg	Ca
Cu, Ag, Au	Sr, Ba

TABLE 2

GENERAL STRUCTURAL INFORMATION CODED FOR ORGANIC COMPOUNDS

SYMBOL	DESCRIPTION
UNSAT-CH [a]	Position(s) of Unsaturation in Chains.
UNSAT-RG [a]	Position(s) of Unsaturation in Rings.
C=C, C=C	Presence of Unsaturation.
STRUCT	Structural Type, e.g., Aromatic.
RINGSIZE	Ring Size.
CHAINLGTH [a]	Chain Length.
NUM-RINGS	Number of Rings.
CHAINSUBS [a]	Position(s) of Chain Substitution.
RINGSUBS [a]	Position(s) of Ring Substitution.
PHYS-STATE	Physical State.
ROTATE	Dextro- or Levro-rotatory.
CIS-TRANS	Cis or Trans Compounds.
SPIRO	Spiro Compound or not.
ALKYL [a]	Type of Alkyl Group Present.
ALKENYL [a]	Type of Alkenyl Group Present.
ALKYNYL [a]	Type of Alkynyl Group Present.
CONJUG [b]	Presence of Conjugation in Compound.
HETEROCYC [b]	Presence of Heterocyclic Compound.

a. Allows multiple coding for more than one entry per compound.

b. Correlated with structural types of tables 3 and 4.

TABLE 3

STRUCTURAL GROUPS (Symbolic Only) CODED FOR ORGANIC COMPOUNDS
CONTAINING OXYGEN, NITROGEN, OR BOTH OXYGEN AND NITROGEN

GROUPS WITH OXYGEN	GROUPS WITH NITROGEN	GROUPS WITH OXYGEN AND NITROGEN
$-C(=O)OH$	$-C(=\overset{|}{N})N<$	$>NC(=O)O-, -OC(=\overset{|}{N})O-$
$-C(=O)O-$	$>NC(=\overset{|}{N})N<$	$>NC(=O)N<, -OC(=\overset{|}{N})N<$
$>C=O$	$-NC$	$-NCO$
$-C(=O)H$	$-CN$	$-C(=O)N<, -C(=\overset{|}{N})O-$
$-OC(=O)O-$	$-NH_2$	$-OCN$
$C(OR)_4$	$>NH$	$>NNO_2$
$-C(OR)_3$	$>N-$	$>NNO$
$-OH$	$>C=N-$	$-NN(=O)-$
$-O-$	$=NN=, >NN<$	$-ONO_2$
$-OO-$	$-N=N-, =N\equiv N$	$-ON=O$
$\geqslant O^+$	$-N=N-N<$	$-NO_2$
$-O_3-$	$-N\equiv N^+$	$-NO$
$>C(-O-)_2$	$\geqslant N^+$	$=NO-, >NO-$
	$-NH_4$	$\geqslant N=O$
	$>NCN$	
	$-N_3$	

TABLE 4

STRUCTURAL GROUPS (Symbolic Only) CODED FOR ORGANIC COMPOUNDS CONTAINING SULFUR, SULFUR AND NITROGEN, SULFUR AND OXYGEN, AND SULFUR, NITROGEN, AND OXYGEN

GROUPS WITH SULFUR	GROUPS WITH SULFUR AND NITROGEN	GROUPS WITH SULFUR AND OXYGEN	GROUPS WITH SULFUR, NITROGEN, AND OXYGEN
-C(=S)S-	>NC(=S)N<, -SC(=Ṅ)N<	-OC(=S)S-, -SC(=O)S-	>NC(=S)O-, >NC(=O)S-
-C(=S)H	>NC(=S)S-, -SC(=Ṅ)S-	-OC(=S)O-, -OC(=O)S-	-C(=O)ṄS-, -C(=S)ṄO-
>C=S	-C(=S)N<, -C(=Ṅ)S-	-C(=S)O-, -C(=O)S-	-OC(=Ṅ)S-
-SC(=S)S-		$-S(O_2)O-$	-S(=O)N<
-SH	-SCN	-S(=O)O-	$>NS(O_2)N<$
-S-	-NCS	-S(=O)S-, -S(=S)O-	$>NS(O_2)O-$
-SS-	>NSN<	S_xO_6 (x=2-6)	$>NS(O_2)-$
$\overline{>}S^+$	=NS-, >NS-	$>SO_2$	>NS(=O)O-
=S:S	-N=S	>S=O	-NSO
	>S=N-	-OSO-	
		$-OS(O_2)O-$	
		-OS(=O)O-	

TABLE 5

ELEMENTS CODED FOR INORGANIC COMPOUNDS

Ac
Al
Am
Sb
A
As
At
Ba
Be
Bi
B
Br
Cd
Ca
C
Ce
Cs
Cl
Cr
Co
Cb
Cu
Cm
Dy
Er
Eu
F
Fr
Gd
Ga
Ge
Au
Hf
Ho
H
In
I
Ir
Fe
La
Pb
Li
Lu
Mg
Mn
Hg
Mo
Nd
Np
Ni
N
Os
O
Pd
P
Pt
Pu
Po
K
Pr
Pm
Pa
Ra
Re
Rh
Rb
Ru
Sm
Sc
Se
Si
Ag
Na
Sr
S
Ta
Tn
Te
Tb
Tl
Th
Tm
Sn
Ti
W
U
V
Yb
Yt
Zn
Zr

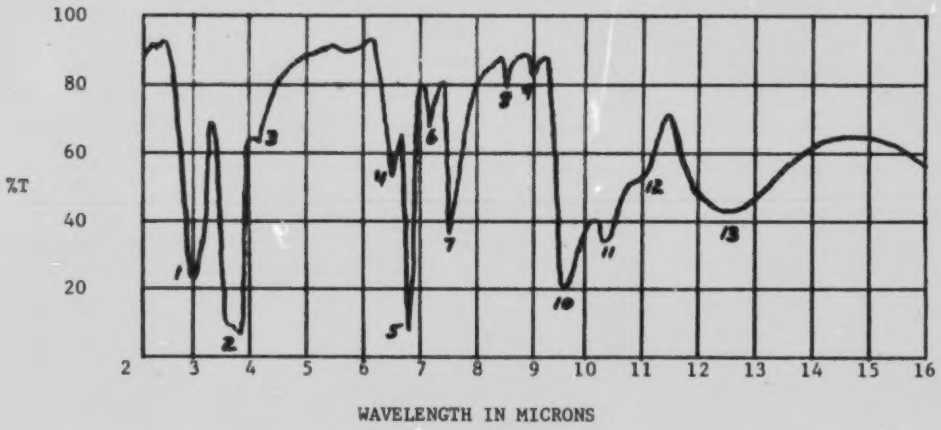
TABLE 6

RADICALS CODED FOR INORGANIC COMPOUNDS

Aluminate
Ammonium
Antimonate
Antimonite
Arsenate
Arsenide
Arsenite
Bismuthate
Borate
Boride
Bromate
Bromide
Carbamate
Carbide
Carbonate
Cerate
Chlorate
Chloride
Chlorite
Chromate
Cyanimide
Cyanate
Cyanide
Ferrate
Ferrite

Fluoride
Fulmanate
Germanate
Hafniate
Hexammine
Hydride
Hydroxide
Iodate
Iodide
Manganate
Molybdate
Nitrate
Nitride
Nitrite
Osmate
Oxide
Pentammine
Phosphate
Phosphide
Phosphite
Plumbate
Plumbide
Rhenate
Selenate
Selenide

Selenite
Silicate
Silicide
Stannate
Stannide
Sulphate
Sulphide
Sulphite
Tantalate
Telluride
Tellurite
Thionate
Titanate
Thorate
Tungstate
Uranate
Vanadate
Zincate
Zirconate
Zirconyl
Platinate
Platinite
Chromite
Gallate
Palladite



Hypothetical Spectrum for Illustration
of Guidelines Necessary for
Coding Spectra

CODED PEAKS - 1, 2, 4, 5, 7, 10, 11, 13

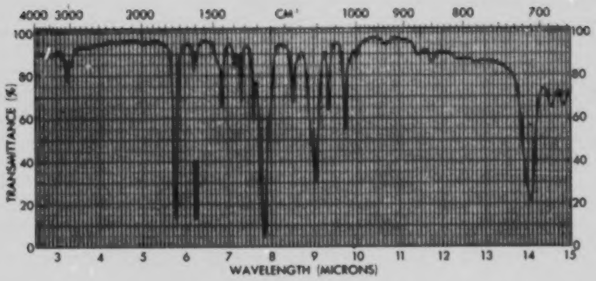
NOT CODED - 3, 6, 8, 9, 12

(CODED AS STRONG - 2, 5)

Figure 1.

Figure 2.

Ethyl Benzoate - 36 hits



INPUT:

```
  RUNTYPE  SPECTRUM=YES
  OPTIONS  WIGGLE=YES
  NOBAND   (3.5,5.5),(10.0,12.0),(12.1,13.7)
  SPECTRUM 5.8,6.9,7.8,9.0,9.3,9.7,14.1
```

OUTPUT:

```
  SEARCH #  2            ORGANIC, HITLIM =  50, WIGGLE,
  RECORDS FOUND 36       INFRARED

  NOBAND REGIONS:   ( 3.5 TO  5.5)   (10.0 TO 12.0)   (12.1 TO 13.7)

  RECORD SERIALS:          1506CA,    1866CA,    2711CA,    8065CA,
      9929CA,    9931CA,   9932CA,    9933CA,   11545CA,   11553CA,
     25038CA,   27233CA,  27674CA,   27770CA,D  1002CA,D   1006CA,
  E   2055CA,E   2316CA,F   178CA,F   446CA,F   558CA,    1429DA,
     10602EA,   11456EA,  11457EA,   13191EA,   1045FA,    2612DA,
      3717GA,     461JA,    583JA,     584JA,    585JA,    3592JA,
      4160JA,    4716JA

  SEARCH SPECTRUM  5.8, 6.9, 7.8, 9.0, 9.3, 9.7, 14.1
```

Five of the 36 hits, 1506CA,F 178CA, 1429DA, 3717GA, 461JA, are recorded as ethyl benzoate. Many of the remaining 31 hits represent closely related esters to ethyl benzoate. Using the PRINT PEAKS OPTION would greatly reduce the problem of analysis of this result.

A QUALITATIVE ORGANIC IDENTIFICATION GAME

Frederick M. Hornack
University of North Carolina
Wilmington, North Carolina 28401

The North Carolina Educational Computing Service under the guidance of a curriculum development manager (Joseph R. Denk) is directing its efforts toward the creation of computer materials for applications in various disciplines. The Triangle Universities Computation Center is the heart of the system and is linked to both batch-handling and direct interaction (conversational) terminals at colleges and universities in the North Carolina network. From the inception the manager has been very active in gathering and transmitting programs to and from other networks.

The chemistry group, consisting of chemistry teachers in large and small institutions, has developed teaching units in two basic modes. The conversational mode, with its rapid and almost human response capabilities, has a natural affinity for teachers. Inspired members of the group have produced several novel teaching programs (as opposed to merely computational programs) in thermodynamics and quantum theory. For example, one program in thermodynamics quickly derives formulas expressing a partial derivative (involving state functions) as a function of experimentally accessible quantities. The user has a choice of about a thousand partials and may verify his own or textbook formulas while perhaps wondering if the computer can do partial differentiation. In another conversational unit, a wave function is plotted and boundary behavior is noted as parameters are varied in the search for an eigenvalue.

Quite early in the work of the chemistry group a vague pessimism began to grow in connection with the use of direct interaction terminals. Even financially favored institutions could afford precious few of these devices. Students were observed waiting impatiently in line and terminals, in some cases, had to be made available in the post-midnight hours. It began to be felt that a batch-type job, preferably with a short turnaround time, would be more generally useful in our network and perhaps others. In this type of teaching unit the student learns by repeatedly submitting his job with varied input parameters of some sort. The output of a run will influence the input for the next run and so on. Obviously, to be successful, the unit must provide some involvement or excitement during the time interval between runs. The well-known business "game" is a good example. The student must make management decisions and sustained interest is generated over a period of perhaps days.

Following the apparent success in the field of business, the curriculum manager and the author (the chemistry group leader) felt that the development of an analogous chemistry "game" of some kind would be a worthwhile effort. Qualitative organic analysis with its varied chemical and instrumental content was the area chosen. The main purpose of this paper is to describe the operation of this identification game (IDGAME).

As we began work on this program, Stanley Smith of the University of Illinois presented a paper (Toronto, May 27, 1970) describing a very sophisticated conversational identification game. The student presents probing questions to the computer and receives rapid answers relating to his unknown compound. Infrared and NMR spectra are displayed at the touch of a few buttons. Each piece of information has a point value. This amazing system (PLATO) is quite different from the simpler one that we have developed. In IDGAME, information has a money value and time for thought and calculation is generally needed between requests for data. In IDGAME, pertinent spectral information is provided rather than a complete display of spectra.

Unknowns and all requests are identified by simple numerical input. Each student request must contain the code number and unknown number for his compound. These numbers (which, for security, internally generate a file number) key the request into a data bank. Each unknown requires about 1000 characters of data in the bank. Required for each compound are spectral data (the main functional IR peaks, the principal mass peaks with isotope information, the NMR multiplicities, intensities, and shifts, and UV data), solubilities, physical properties, visual test results, quantitative data (e.g., % composition, saponification number, etc.), derivative melting points, and preliminary description (odor, etc.).

Normally the student first requests by the digit zero a printout of all available information. This will appear as shown on the following page.

```
STUDENT NAME:  SMITH
*** UNKNOWN NUMBER  15 ***

                  *** AVAILABLE INFORMATION ***

  1 DESCRIPTION OF UNKNOWN

PHYSICAL PROPERTIES

  2 BOILING POINT
  3 MELTING POINT
  4 REFRACTIVE INDEX(SODIUM LIGHT)
  5 SPECIFIC GRAVITY T/4
  6 OPTICAL ROTATION DATA(SODIUM LIGHT,2 DECIMETER TUBE,CONC G/ML)

ANALYSIS

  7 MOLECULAR WEIGHT DATA(VAPOR DENSITY OR F.P. DEPRESSION)
  8 TITRATION OF 1.00 GRAM WITH STANDARD BASE(NEUT. EQUIV.)
  9 HEATING 1.00 GRAM WITH EXCESS BASE AND BACK-TITRATING(SAPON. EQUIV.)
 10 ELEMENTS BY SODIUM FUSION
 11 QUANTITATIVE ANALYSIS FOR CARBON
 12 QUANTITATIVE ANALYSIS FOR HYDROGEN
 13 QUANTITATIVE ANALYSIS FOR NITROGEN
 14 QUANTITATIVE ANALYSIS FOR SULFUR
 15 QUANTITATIVE ANALYSIS FOR PHOSPHORUS
 16 QUANTITATIVE ANALYSIS FOR HALOGEN

SOLUBILITIES

 17 SOLUBILITY IN WATER
 18 SOLUBILITY IN DIETHYL ETHER
 19 SOLUBILITY IN DILUTE SODIUM BICARBONATE
 20 SOLUBILITY IN DILUTE SODIUM HYDROXIDE
 21 SOLUBILITY IN DILUTE HCL
 22 SOLUBILITY IN COLD CONCENTRATED SULFURIC ACID

CLASSIFICATION TESTS

 23 VISUAL EFFECT ON AGITATING WITH ZINC CHLORIDE-HCL SOLUTION(LUCAS)
 24 VISUAL EFFECT ON ADDING TOLLENS' REAGENT(AGNO3-NH3)
 25 VISUAL EFFECT ON MIXING WITH NAOH,BENZENESULFONYL CHLORIDE,AND ACIDIFYING(HINSBERG)
 26 VISUAL EFFECT ON SHAKING WITH ETHANOL AND 2,4-DINITROPHENYLHYDRAZINE
 27 VISUAL EFFECT ON ADDING DILUTE POTASSIUM PERMANGANATE TO ALCOHOLIC SOLN.
 28 VISUAL EFFECT ON SHAKING DIOXANE SOLN. WITH NAOH AND IODINE-KI SOLN.(IODOFORM TEST)
 29 VISUAL EFFECT ON SHAKING WITH FUMING SULFURIC ACID
 30 VISUAL EFFECT ON WARMING WITH AZOXYBENZENE AND ALUMINUM CHLORIDE

DERIVATIVES

 31 SOLID PRODUCT ON HEATING WITH THIONYL CHLORIDE AND ADDING ANILINE
 32 SOLID PRODUCT ON HEATING WITH ALPHA-NAPHTHYL ISOCYANATE
 33 SOLID PRODUCT ON WARMING WITH NAOH AND BENZENESULFONYL CHLORIDE,AND ACIDIFYING
 34 SOLID PRODUCT ON HEATING ALCOHOLIC SOLUTION WITH PICRIC ACID
 35 SOLID PRODUCT ON TREATING WITH 2,4-DINITROPHENYLHYDRAZINE REAGENT
 36 SOLID PRODUCT ON HEATING WITH 3,5-DINITROBENZOIC ACID AND ACID CATALYST

SPECTRAL DATA

 37 INFRARED
 38 ULTRAVIOLET
 39 MASS
 40 NUCLEAR MAGNETIC RESONANCE
```

It can be seen later that the phrases have been specifically designed to integrate with output messages. Larry Sherman of our network has assigned realistic laboratory costs to each of the forty items. His time and cost sheet is included as a hasty appendix to this paper. The object of the game is to identify the unknown organic compound at minimum cost.

Specific requests are made until identification is complete. The cautious student will request only a few items at a time and will weigh the output carefully. Calculations may be needed, for example, in connection with requests for rotatory power, molecular weights, and saponification numbers. It will be seen below that in such cases raw data rather than final results are given. Time is also needed to check percent composition.

For each item, verbal and numerical replies have been worked out. For program economy some of the phrases used in the listing of available information are used in the output format. In the examples discussed below, the student name and unknown number, which always preface the chemical output, have been omitted.

A student requesting the Lucas treatment (number 23) and derivative information (numbers 32 and 33) may receive the output:

```
23 *** LAYER IN 10 MIN ON AGITATING WITH ZINC CHLORIDE-HCL SOLUTION (LUCAS)
32 *** URETHAN, MP 105-107 C ON HEATING WITH ALPHA-NAPHTHYL ISOCYANATE
33 *** NO LUCK ON WARMING WITH NAOH AND BENZENESULFONYL CHLORIDE, AND
ACIDIFYING
```

"LAYER IN 10 MIN", "URETHAN, MP 105-107 C", and "NO LUCK" are the data portions of the output.

Examples of output requiring numerical calculation follow:

```
8 *** TITRATION OF 1.00 GRAM WITH STANDARD BASE (NEUT.EQUIV.):  13.2 ML .1N NAOH

11 *** 59.96% CARBON
```

Further examples of output follow:

```
1 *** DESCRIPTION OF UNKNOWN:  NUMBER 31 IS A COLORLESS LIQUID WITH A
PEAR-LIKE ODOR.

37*** INFRARED DATA
DIAGNOSTIC PEAK LOCATIONS (MICRONS):
3.28 3.37 5.73(S) 8.15(S,B) 13.35(B) 14.35(B)

39 *** MASS SPECTRAL DATA
PROMINENT PEAK LOCATIONS(M/E):
15,18,27,45,59
ISOTOPE ABUNDANCES RELATIVE TO PARENT PEAK(M/E AND % OF P):
58(P) 100,59 3.5,60 .18

5 *** SPECIFIC GRAVITY T/4: 20/4, 0.792
```

Early trials at local computer workshops have indicated some enthusiasm. At the time of writing this paper it is planned to complete documentation to aid in the use and expansion of the unit.

Prior to the presentation of this paper several changes will have been made. Two subroutines are being added to calculate simplest empirical formulas from percent composition and to generate mass spectral isotopic abundance ratio tables for specific mass numbers. Several "consultation services" and an ammonia odor test on hydrolytic heating are being added. Costs will be assigned to these items.

It is hoped that by meeting time it will be possible to report the results of student trials.

APPENDIX

Qual Unknown Game

		Hours	
1.	5 min.	.1	$2
2.	24 min.	.4	8
3.	12 min.	.2	4
4.	12 min.	.2	4
5.	42 min.	.7	14
6.	1 hour	1	20
7.	3 hours	3	60
8.	12 min.	.2	4
9.	48 min.	.8	16
10.	24 min.	.4	8
11.	20 (CHN analyzer)	.3	12
12.	20	.3	12
13.	20	.3	12
14.	1½ (SO_4 turbimetric or gravimetric)	1½	30
15.	3 hours (PO_4- colorimetric)	3	60
16.	48 min. (titrate)	.8	16
17.	5 min.	.1	2
18.	5 min.	.1	2
19.	10 min.	.2	4
20.	5 min.	.1	2
21.	5 min.	.1	2
22.	5 min.	.1	2
23.	12 min.	.2	4
24.	18 min.	.3	6
25.	18 min.	.3	6
26.	18 min.	.3	6
27.	15 min.	.3	6
28.	25 min.	.4	8
29.	12 min.	.2	4
30.	18 min.	.3	6
31.	(1½ hr.)	1.5	30
32.	(1½ hr.)	1.5	30
33.	(1½ hr.)	1.5	30
34.	(1½ hr.)	1.5	30
35.	(1½ hr.)	1.5	30
36.	(1½ hr.)	1.5	30
37.	30 min.	.5	10
38.	24 min.	.4	8
39.	1½hr.	1.5	30
40.	2½ hr.	2.5	75

COMPUTER USAGE IN CHEMICAL EDUCATION AT THE MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Paul L. Houston, Jeffrey I. Steinfeld and David G. Sutton
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

For the past two years we have been engaged in integrating into the chemistry department curriculum at MIT an interactive on-line computer facility under the sponsorship of the National Science Foundation. Our efforts to date have been in three major categories: (1) the development of packaged interactive programs for interactive analysis of laboratory data by students (including graphical display), (2) the implementation of the OMNITAB system developed at the National Bureau of Standards as an alternative to conventional FORTRAN programming, and (3) production of computer animated films. Brief reports on each of these areas will be given. The system which we have used in this work is the CP/CMS system developed at MIT. This system is based on an IBM 360/67 computer. Access to CMS is accomplished by means of a 2741 type terminal. The programs have all been written in FORTRAN 4, thus enabling them to be transferred to any computer terminal such as 1130 or 1620 which has an interactive keyboard.

(1) Interactive programs for student analysis of laboratory data.

Many otherwise useful laboratory experiments are not included in the usual repertoire of student experiments primarily because the data analysis involved in their interpretation is much too tedious and time consuming. This, of course, is where the availability of computation facilities can make all the difference. In addition the use of the computer in this way serves to both motivate and introduce the student to its use. We have developed a number of programs which are stored in our system for the analysis of data obtained in pre-existing experiments in our laboratory curricula. We expect that the availability of the computer will lead to the introduction of new experiments which could not have been contemplated until this facility was available. Some of the experiments which we analyze in this way are: (a) a complete package for analysis for x-ray powder patterns for all crystal groups is available, (b) spectroscopic analysis of the infrared spectra of HCl and DCl is carried out with the use of a generalized least square fitting procedure which enables the student to weight and select his data according to the goodness of fit at each stage, (c) an experiment is carried out on the heat capacity of solid argon, and (d) the kinetics of the enzymatic inversion of sucrose is analyzed. We are at present supplementing our teletypewriter facilities with a graphics terminal (the Advanced Remote Display System based on the Tektronix 611 oscilloscope). This is just now being brought on line and results of its use will be reported at the conference.

(2) OMNITAB programming system.

As an alternative to FORTRAN programming we have implemented the OMNITAB programming system developed by Joseph Hilsenrath at the National Bureau of Standards. This system embodies an extensive library of statements which treat data in tabular form. The outstanding feature of this system is that it permits calculations to be carried out one step at a time, with the results obtained immediately before the next step is carried out. This eliminates the editing and compilation steps in ordinary methods of programming. The OMNITAB package also includes easily accessed, comprehensive curve-fitting, statistical analysis, and plotting subroutines which are of general use to students and research personnel.

(3) Computer animated films of molecular collisions.

As an adjunct to the work described previously, we have been engaged in producing computer animated motion pictures of molecular collision trajectories, using the programs originally developed by M. Karplus and co-workers at Columbia University and Harvard University. The purpose of such films is both to enable an investigator to visualize the dynamics of a reactive or inelastic collision, and also to build up a library of educational materials which can be used to demonstrate graphically the nature of molecular collisions for beginning courses in chemistry. The particular systems for which calculations have been carried out involve vibrationally and rotationally inelastic collisions of excited diatomic molecules. We plan to show several films as part of our presentation.

Acknowledgments

We wish to thank Miss Diane Feldman, Dr. Philip Manor, Mr. Andrew V. Nowak and Mr. Michael Rubinson for their assistance on this project. This work was supported by Grant GJ-350 from the National Science Foundation.

REPORT ON
AN ATTEMPT TO UTILIZE COMPUTERS
IN URBAN AFFAIRS EDUCATION

Brian S. Vargus and Douglas White
University of Pittsburgh
Pittsburgh, Pennsylvania

It is not unusual for the undergraduate student majoring in urban affairs or taking courses with urban problems as their focus to come in contact with computer simulations, computer games or role playing games. However, in virtually all of these experiences the student is confronted with a particular and special set of alternatives determined by the game or the computer simulation. In large part, these simulations or computer games are based on the student taking the role of a city councilman, city planner, mayor or some other figure in the political "power structure." This is not without its merits and has provided many students with positive experience in simulated leadership situations. However, it is also possible that experiences in leadership games or economic growth model simulations lead the student to view the city only from the perspective of the political or economic leadership. (1) Such experiences seem to reinforce the charge increasingly made by both government and industry that college students with undergraduate training are ready for leadership positions but are not willing to take a job of narrower scope or to become involved with problems at less than a macro level. On the other hand, in urban sociology and urban anthropology there is a strong tradition of research on neighboring and informal social relations within geographically small units. (See Keller, 1968) This tradition has pointed out the difficulties of defining something like a neighborhood, but has also stressed the strengths of communal bonds in these relatively small geographic areas. It has further suggested that there are human resources in these areas that can be mobilized to attack various kinds of problems now facing urban areas. Indeed, one of the strongest cries we have today is for more citizen participation in government programs through greater decentralization. (See Altshuler, 1970)

With these facts in mind, the authors designed and taught a course at the University of Pittsburgh for undergraduate students which attempted to have them deal with the city from a "bottoms-up" point of view. The basic notion was that it is more important for students to realize that more goes on in cities that influences human lives than city council meetings, boards of directors meetings, and city planning hearings. It was felt that it is important to counter the image which is widespread in the social sciences that the city is an impersonal heterogeneous kind of environment in which people have little regard for their neighbor and in which people are highly alienated from one another. (2)

With the goal in mind of making students aware of informal social relations a computer simulation was devised based on ten input variables that could be easily observed by a layman who had been introduced to some simple concepts in participant observation and interviewing. This simulation was designed to produce a profile of neighborhood social interaction indicating the quality and locus of informal social interaction in a neighborhood, neighborhood relations with and attitudes toward public bureaucracies, forms and degrees of tension in the neighborhood, and the ability of the neighborhood to respond as a unit to any crisis. The program itself was quite simple. The intent, however, was rather complex. First, it was intended to make students aware that most of urban life is informal behavior that is far removed from things like city council budgets, though it is certainly influenced by those things. Second, we hoped to show it is possible to simulate, on a computer, the quality and form of informal social relations in a neighborhood. Most students indicated a strong skepticism about the value and possibilities of computers simulating less "grandiose" patterns of behavior than legislative decisions, economic growth of an entire system, etc.

To this point, there is nothing unusual in the use of the computer in the course of instruction. The innovation, however, which we found most useful as a pedagogical tool and in our research was to actually send the students out to collect data to feed into this simulation model, allowing them to evaluate the output statements of the simulation and to suggest changes in order to produce a more accurate neighborhood social interaction profile.

The students were first introduced to the general topic and procedure of simulation. The strategy and objectives of simulation were outlined using Raser's (1969) Simulation and Society. Students were asked to play a somewhat frustrating role playing game where each took the part of a member of the Mayor's Commission on Human Relations dealing with an interethnic conflict in the city. Both interpersonal and group conflicts were maximized in the game. A number of the roles in the initial "game" were built around indigenous neighborhood organizations and leaders. From there the students were introduced to the notion that cities are more than mayors, city councils, and commissioners. A brief discussion was given of the concept of neighboring and neighborhoods and a set of readings was assigned to sensitize the students to informal social relations in the urban environment. (3) At this point, the computer model was presented to the students. It was presented as a simple model which on the basis of ten rather clear variables would produce a profile of social interaction in any neighborhood.

The program was written in FORTRAN IV for an IBM 360-50 system with time sharing. Students could, therefore, go to a remote console and using the job control language initiate the execution of the program. Each student was shown how to use the remote console and how to load the particular program, called URBAN, to produce the neighborhood social interaction profile. Students were encouraged to input, by following the very precise instructions given within the program, ten variables about their own home neighborhoods--that is, their neighborhood before they came to the University. At this point they were not asked to evaluate the outputs but were merely given experience in using the computer and using the particular model developed by the instructors. Comments from the students ranged from very positive to very negative. Some students suggested the model was totally wrong while others found it to be "marvelously accurate." The students were stimulated to ask many questions. "How did the computer arrive at the output statements?" "What kinds of 'equations' were used to determine the outputs?" "What sort of theory did you use to develop the model?" In each case the students were referred to a comprehensive reading list for various examples of studies that had hypothesized or demonstrated relationships between the input variables we used (mainly socio-economic kinds of things) and neighboring relations.

The program was based on three very simple assumptions. First, ecological factors influence neighboring relationships. (See Michelson,1970; Suttles, 1968) Second, socio-economic status influences neighboring relationships. (See Laumann, 1966) Third, life style and stage of family cycle influence neighboring relationships. (See Gans, 1966) Other input variables were also used though these three carried the greatest weight. The other variables included: ethnicity, religion, stability of the neighborhood, traffic on the streets, and apparent quality of input of public services. The students were instructed rather carefully on how to arrive at an accurate estimate of each of these through observation (e.g., condition of houses for S.E.S.) and the use of key informants (e.g., realtors for stability). From these the neighborhood interaction social profile was generated. (4) The output profile was based on six general things. First, it indicated the degree of openness of the neighborhood to outsiders. Second, it gave an evaluation of the neighborhood's reaction to various kinds of social services, i.e., the schools, the police, and other public services. Third, it gave a description of the locus of interaction in the neighborhood. Fourth, it gave an indication of the degree to which neighborhood social interaction was primary or secondary and a description of the quality of each. Fifth, it gave an indication of the ability of the neighborhood to respond to crisis and the likelihood that residents could rely on their neighbors in times of crisis. Finally, it gave an indication of the degree and form of tensions existing in the neighborhood.

After the students had utilized the program by entering variables from their home neighborhood they were assigned an area of approximately two blocks in the city of Pittsburgh to observe. (5) They were told to observe the same ten input variables they had used for their own neighborhood, i.e., socio-economic status, religion, traffic, distribution of dwelling units, land use patterns, etc. This information was fed by the students into the computer identifying it as real data. (6) This activated an option in the program which required the student, at the end of the run, to evaluate the accuracy of the output statements and to evaluate the degree to which he felt his input data were accurate. These evaluations were stored by the computer for later use by the designers of the model in their revisions. In other words, the students actually became participants in refining the computer model and in refining the kinds of equations used in generating the neighborhood informal social interaction profile. Multiple evaluators were thus added to the feedback loop essential to refining a model. Subsequent analysis of the evaluations has allowed us to begin extensive revisions in the model to make it more accurate.

The field observation exercises were received quite positively by the students. Students reported themselves as becoming "quite involved in their neighborhood." Many students said they wished they could have spent more time observing their neighborhood and getting to know the people in it. To this point, the students had merely become sensitized to informal interaction in the neighborhood and the existence of the neighborhood in an urban environment. It was with the evaluation of the computer model that students began to see the usefulness of this kind of experience.

Most students found something wrong with at least 20% of the output statements in their simulation. They were stimulated to suggest changes in the model so that the authors could improve the predictions for the particular neighborhood the students had observed. This resulted in a large number of papers from the students showing a new appreciation for informal social interaction in an urban context and the influence of all sorts of variables on neighboring. Beyond this, the papers showed a surprising awareness of the potential usefulness of a simulation model in understanding neighborhood dynamics. Many of the students pointed out that they had not been aware of the importance of neighborhoods. Others suggested that they could now see a rationale for arguments for participation by people in urban government and the decentralization of governmental authority. Still others felt a new concern for urban problems and the human dimension of such problems. For example, one student wrote,

> "In reading about urban neighborhoods, the processes of interaction never became apparent to me. I had a theoretical knowledge of the urban neighborhood but I had never looked into the processes myself. The studies I read were clear. The methods I studied for doing research were understood. Somehow, they remained somewhere in a theoretical realm, far away from my experience. The program as a whole predicted rather accurately the realities of the neighborhood. The criticisms I raised deal mainly with attempts to make the program more accurate when dealing with our neighborhood. The urban community is a very complex system and an attempt to systematize relationships between input variables and output predictions would result in many inaccuracies when examining small, specific neighborhoods. The program succeeds in providing a general framework for observation."

Another student writes about her new sense of what a neighborhood is.

> "Before taking this course I never thought very much about what a neighborhood really was. I always had some vague conception in my mind of a neighborhood being a section of a town or city defined by some physical boundary such as streets and made up of a certain number of people working as a cohesive unit. Then you made us talk of the neighborhood in which we grew up. In the middle of the course I was given a section of Bloomfield to observe. The most interesting part of these observations was comparing first observations with real data acquired from key informants and comparing them back to the program model. With some corrections I feel the model will work. In general, I would have to conclude that if there were no flaws in the input material the program would have made a fairly accurate prediction."

However, not all students' comments were positive. One student after indicating severe reservations about the model, indeed, what he called "serious flaws" wrote:

> "I don't know if you can correct your model, or if you can, if the effort is worth it. In order to gather data for the required input statements, it is necessary for a researcher to spend some time in a neighborhood. While there, he will almost certainly observe some of the neighboring patterns, which will make unnecessary the observations of the computer. The computer's predictions are so broad and insensitive to special circumstances, that the increased time spent by a researcher in observation and inquiry will be more valuable than the output statements of the computer."

Still another student writes,

> "Other than urban planning and community control, I can distill no other implications for such a model, but could appreciate even an elaboration on these two points. The class seems to have ignored any such explanation and thereby what might have been a more passionate involvement dwindled to a level of boredom. It is important to know why you are doing something if you value your life as meaningful. In terms of the computer model, all I can suggest is to build it into a larger system which somehow might incorporate sense of self, social consciousness variables, natural leadership variables, etc., to clarify the degree of identity residents feel towards each other and their ability to act as an organized unit."

The final sessions of the class were spent doing two things. First, in response to critiques such as the last two, the students and instructors engaged in a dialogue about the usefulness of the model. Considerable discussion was generated about the resistance to attitude surveys, particularly in low income areas (see Josephson, 1970), and the possibilities for an observational model to generate neighborhood profiles that might aid planning agencies, welfare agencies, etc. Some students noted that a few simple observations plus data already available to an agency such as that in the census could reduce the costs incurred through massive survey projects. (7) Second, the students suggested many more changes in the model. One student noted that he felt he had learned a great deal by simply "playing with the model," i.e., constructing hypothetical neighborhood situations and then informally evaluating the neighborhood social interaction profile which was produced. Some "made sense" to him but others were "really weird." Thus, through the use of a computer simulation in which students were asked to critique the program after entering real data collected by themselves, they have developed increased awareness of the informal dimension in urban social life and some new possibilities for the uses of computers.

This paper has described the use of a computer simulation of informal neighborhood social interaction combined with student field work in order to sensitize students to the less "glamorous" aspects of urban life. On the basis of our experience with this model the authors feel that it has broader implication as an urban planning instrument as well as an instructional tool. It seems clear to us that the use of a computer simulation as a challenge to students, that is, asking them to attempt to validate or reject the model, is a very useful tool. It is apparent to us that our students were sensitized to a new dimension in urban life, a dimension that is too often overlooked in the standard textbook describing problems in urban areas. Further, the students' experience in validating a simulation through their own field work seems to have developed in them a new appreciation for the usefulness of computers in aiding in the solution of various kinds of human problems. It is apparent that many of the students see that it would be useful for a city planning agency to have a model such as the one developed here, albeit somewhat more accurate, thereby avoiding the necessity of expensive, obtrusive and time consuming survey research in a "target" neighborhood before planning changes such as redevelopment, rezoning, new public construction, etc. Since many of these students plan careers in fields such as urban planning and urban administration the computer usage in this course seems to have increased their awareness of urban social life in a way that could produce significant changes in the actions of many municipal agencies.

BIBLIOGRAPHY

1970 Altshuler, Alan, Community Control: The Black Demand for Participation in Large Cities. New York: Pegasus.

1969 Forrester, Jay W. , Urban Dynamics. Cambridge, Massachusetts: M.I.T. Press.

1962 Gans, Herbert, Urban Villagers. New York: Free Press.

1966 Gans, Herbert, "Urbanism and Suburbanism as Ways of Life: A Re-evaluation of Definitions," in Arnold Rose, Editor, Human Behavior and Social Process. Boston: Houghton-Mifflin.

1970 Josephson, Eric, "Resistance to Community Surveys," Social Problems, 18 (Summer): 117-129

1968 Keller, Suzanne, The Urban Neighborhood, New York: Random House.

1966 Laumann, Edward O., Prestige and Association in an Urban Community. Indianapolis: Bobbs-Merrill.

1970 Michelson, W.H., Man and His Urban Environment. Reading, Massachusetts: Addison-Wesley.

1968 Morris, R. N., Urban Sociology. New York: Praeger, Co.

1969 Raser, John R., *Simulation and Society: An Exploration of Scientific Gaming*. Boston: Allyn and Bacon, Inc.

1968 Suttles, Gerald, *The Social Order of the Urban Slum*. Chicago: University of Chicago.

1966 Webb, Eugene, Campbell, D. T., Schwartz, R.D., Sechrest, Lee, *Unobtrusive Measures: Nonreactive Research in the Social Sciences*. Chicago: Rand McNally.

1938 Wirth, Lewis, "Urbanism as a Way of Life," *American Journal of Sociology*, 44 (July), pp. 1-24.

FOOTNOTES

1. See, for an example, Forrester (1969)
2. This image stems very much from the influence of Wirth's (1938) paper, "Urbanism as a Way of Life." For an excellent critique, see R. Morris (1968).
3. Gans (1962) was particularly useful here.
4. The attached "Observation Record" sheets indicate the variables which were used as inputs and outputs. It is a copy of the forms given students to aid in their fieldwork. Students were also asked to keep notes on any interviews with 'key informants.'
5. The choice of two blocks was an arbitrary one. The size of a neighborhood is extremely hard to ascertain. As it turned out some of the students reported finding two or three "neighborhoods" in the area assigned them while others reported they only had "part" of the neighborhood.
6. In those cases where students found more than one "neighborhood" in their area they were asked to make a separate entry for each. Those reporting they only found "part" of the neighborhood were instructed to treat what they observed as a whole unit.
7. The strategy developing seems to be based on asking for "less obtrusive" measures. See Webb, et al (1966).

Observation Record

Outputs: Briefly describe

A. Openness and degree of acceptance of Outsiders:

B. Attitude towards and acceptance of Police:

C. Attitude towards and acceptance of Public Schools:

D. Attitudes towards and acceptance of other Public Services (e.g., welfare, street department, general governmental institutions):

E. Locus of Neighboring or Informal Interaction (e.g., in home, on porch, on street, pubs, etc.):

F. Percentage of Neighborhood Interaction that is informal or "elementary" social behavior and its intensity (e.g., very intimate, merely perfunctory greetings, etc.):

G. Probability that personal (family) or local "crises" will be responded to by "neighborhood" action:

H. Percentage of Formalized neighborhood interaction (i.e., formal groups and associations--Number and form of participation in such groups and their influence on "neighborhood" spirit):

I. Degree of Neighborhood Tensions (i.e., amount of conflict and availability of mechanisms to resolve conflicts):

COMPUTER APPLICATIONS IN THE UNDERGRADUATE SOCIOLOGY CURRICULUM AT OBERLIN COLLEGE*

Stephen J. Cutler and David V. Blagg
Oberlin College
Oberlin, Ohio

Introduction

Unquestionably, there is now a trend in the social sciences in the direction of providing an early introduction--often in the beginning course--to the quantitative aspects of the several disciplines. On the one hand, this trend reflects the increasing presence of computer facilities on many campuses; on the other hand, it represents a desire to improve the undergraduate educational process. However, this new emphasis is not being achieved without certain problems in bringing the resources of the computer into the classroom.

The nature of these problems is most apparent if we insist that an introductory course should remain a general introduction to a discipline, most often taken by students early in their undergraduate career. In order to minimize the level of technical expertise needed for students to utilize the computer, there is a press toward a large computer installation, generously supported and well equipped in both hardware and software. Yet in the small college or university, there are frequently insufficient resources to maintain such a sizeable facility. And if the resources with which to simplify instruction are unavailable, the goal of acquiring elementary analytical skills must contend with the necessity of first learning extensive computer oriented technical skills. In these terms, the major problem is that if there are constraints on hardware and software, the knowledge and skills required for a meaningful introduction to analytical methods are inappropriate for the beginning social science course. Put still another way, the problem resolves itself into the following question: how can a small college with limited computer resources provide instruction in computer-based analytical methods in its introductory social science courses without requiring undue technical knowledge?

In the Department of Sociology-Anthropology at Oberlin College, we have been considering this question over the past two years while attempting to create and to collect materials appropriate for the teaching of elementary analytical and methodological techniques in the beginning sociology course. We proceeded under certain assumptions and constraints which, because they are necessarily reflected in our end result, should be stated at the outset. To begin with, we assumed that first-hand knowledge of sociological research techniques, at however an elementary level, would give students a powerful tool with which to frame and to pursue questions of their own interest beyond the usual vehicle of library research. Second, we assumed that the acquisition of analytical and methodological techniques which utilize the computer would allow students to approach sociology from a more informed and sophisticated perspective. Third, we assumed that for the majority of the introductory sociology students such exposure to the computer as they might have would be their first. Because of this, and because our orientation was largely limited to the learning of analytical skills, we felt it critical that there be no need to learn an extensive programming system (such as FORTRAN) to enable data manipulation.

Given these assumptions, we were also reluctant to introduce the students to data gathering techniques in any more than a cursory manner. In other words, we maintained that the refinement and extension of elementary analytical skills is more properly the domain of middle and advanced level methods courses. Therefore, since primary analysis (in the usual sense of the term) was ruled out, we settled upon secondary analysis as the principal research vehicle, albeit with the explicit mandate that the data provided would be of clear theoretical and substantive sociological interest but with sufficient scope and flexibility so that the diverse interests of the students who would be using the data could, in most cases, be met. In short, the development of the materials and methods presented in this discussion was guided by the general assumption that both the teaching and learning of sociology would be considerably enhanced and complemented by the students "doing" sociology. At the same time, we insisted that for the first such exposure, this introduction must be characterized by ease of understanding and simplicity of use combined with many attractive alternatives to engage the variety of substantive interests of the students.

With these assumptions in mind, we have put together a brief curriculum for the introductory sociology course which makes use of Oberlin's IBM 360/44 computer. (1) We intended that the initial experiences in using these materials in the fall of 1970 would lead to refinement and revision, followed by a more rigorous evaluation of their impact. However, our perception of a very favorable student and faculty response to their availability suggests that some discussion of our efforts will be helpful to others who are moving or who are about to move in this direction. Toward this end, we will discuss the nature of the curricular materials, their use in the introductory course and, while unanticipated at the time the project was formulated, in the research methods course, and some conclusions concerning our experience and probable directions for improvement.

Materials and Procedures

Our specific intentions were two-fold: first, we wanted to develop materials that would introduce the students to the utilization of the computer; second, and equally important, we wanted to develop materials that would provide an orientation to the theoretical and operational formulation of research problems, the interpretation, rendering, and presentation of output, and the meaning of both theoretical and statistical relationships between variables. In the most general sense, then, the materials were designed to give an overview of the research process and a sufficient introduction to its mechanics so that students could move easily and quickly to more in-depth analyses according to their interests. In what follows, we present a brief description of these materials and their use in the classroom.

Instructional Materials

Experience has pointed to the general uneasiness, if not genuine fear, of many students when confronting the computer for the first time. For this reason, we felt it imperative that their introduction to the computer be closely guided and lead quickly to a sense of confidence by assuring the success of the initial runs. To accomplish this, we revised, simplified, and illustrated the documentation of the general purpose computer program (2) sponsored by the Sociology-Anthropology Department and thus created our basic instructional document, AN INTRODUCTION TO DAFPACK.

To characterize the content of our document, its scope is limited to only two of the many statistical and analytical options available in the general program--frequency distributions and bivariate and multivariate tabular analysis. More specifically, a relationship in the general area of stratification and socialization is taken from the sociological literature to serve as a substantive framework. The document then proceeds to discuss the utilization of the DAFPACK program to replicate this relationship almost exactly using more recent data from one of our own data sets. The general purpose of each system and program control card is discussed along with a specific designation and explanation of what is to be punched on each of the cards for the purpose of this particular run. In other words, the discussion attempts to give the students a general introduction to the use of the program as well as a concrete application.

Following the more extended discussion of the mechanics of obtaining this particular relationship, the entire card set-up is reproduced so that the students need simply to punch and to submit the cards and then compare their output with a table giving a complete substantive rendering of the results they would get. In precisely the same detailed manner, the document continues on to consider the mechanics and utility of controlling for a given variable, of recoding or collapsing variables, and of examining frequency distributions. Thus, a total of four carefully structured runs for the students to submit are presented.

A second instructional document, THE RELATION OF THEORY TO SOCIAL RESEARCH, traces movement from the theoretical notions of alienation and anomie to the operational notion of powerlessness. As exercises, students are asked to identify the conceptual origins of a number of standard items commonly used to measure alienation, anomie, and powerlessness, to identify those items intended to measure powerlessness in the codebook for the Powerlessness data set, and to obtain and to discuss at least one relationship employing powerlessness as either an independent or dependent variable. Through these exercises, then, the students are given a more structured theoretical orientation to research problems but less guidance in the actual utilization of the computer to consider these problems.

Two additional documents were prepared to supplement the major instructional materials. One, ON CONSTRUCTING AND READING TABLES, notes some general considerations in constructing and interpreting tabular presentations. In brief, we include here aspects such as the general structure and content of tables, rules on the direction of percentaging, and both the reading and substantive explication of some examples of bivariate and multivariate tabular relationships. The final document, CHI-SQUARE, was designed to discuss the relationship between theoretical and statistical significance and introduces in a most elementary manner concepts such as observed and expected frequencies, the null hypothesis, and degrees of freedom. Simple examples of the computation and interpretation of chi-square as a test of significance are also given. A few brief exercises require the students to compute and to interpret the chi-square value of a relationship given in the document and to discuss the significance of the results of some chi-square values obtained using the computer. This particular document is especially appropriate since summary measures of association and chi-square values are part of the DAFPACK output obtained from the exercises contained in most of the documents.

Data Sets

In order to maximize the range of interests that could be pursued by students, we constructed a number of data sets with appropriate codebooks. (3) They were created in such a manner that all would offer a fairly complete and extended selection of variables, but none would be so large and unwieldy that students would be unable to handle them or excessive machine time would be required to process them. For simplicity, only one--the Powerlessness data set--is utilized in the instructional materials.

Some indication of the actual range of the substantive topics in the data sets is suggested by the partial descriptions given below:

1. Police Data--information on social background; measures of attitudes toward minority groups, the use of force, changes in police roles, the professional status of the police, and problems faced by the police.

2. Student Data--from a study of students at nine colleges, with measures of attitudes toward civil rights and anti-war activities; political, religious, and ethical attitudes; relevance and importance of academic disciplines; and background information.

3. International Development Data--measures of economic, social, and demographic development for 142 nations.

4. Urban Families Data--from a study of urban blacks, with measures of family composition, life-style, solidarity, problems, and financial resources; patterns of participation and interaction; aspirations for children, and background data.

5. Powerlessness Data--from a study of the correlates and consequences of powerlessness with socio-economic data; patterns and extent of social interaction; attitudes about family structure, labor unions and government; and powerlessness measures.

6. Fertility Data--from a study of factors affecting family size, with measures of family composition and marital characteristics; religious affiliation and values; socio-economic information; and attitudes about family authority structure.

7. Voluntary Association Data--from a study of the correlates and consequences of voluntary association membership, with data on membership and involvement in types of associations; socio-economic data; and attitudinal measures of powerlessness, dogmatism, tolerance of ideological nonconformity, and status concern.

8. Senior Citizen Data--with background and socio-economic information; attitudinal information on powerlessness, social isolation, life satisfaction, and political intervention; and patterns and frequency of social interaction.

To facilitate student familiarity with the contents of the several data sets, another document was prepared which gives the formal eight-character data set name as catalogued in the computer library, the mode of storage (cards, tape, or disk), a brief comment on the origin of the data set, and a description of the major substantive topics included in each data set.

Aside from these specifics, a number of points can be made about the utility of stored data sets. Clearly, with the exception of the one upon which all of the curricular materials are based, any data set which proves to be unmanageable for students can be removed readily. On the other hand, as new data sets become available, they can just as readily be added to the existing library. Also, as is implied by the brief descriptions above, such data sets provide the students with a rich selection of variables from which to choose. It is worth underscoring this point in another way: while any given data set may have been created originally for a given analytical purpose, the range of variables included is usually such that analyses far afield from the original intention can be undertaken easily. Viewed this way, the average of 120 variables on each of the eight data sets permits literally hundreds and hundreds of possibilities. Finally, the number and range of data sets is not limited by the particulars of a packaged program but rather by the resourcefulness and imagination of whoever assembles them.

Classroom Instruction

Assuming the desirability of acquainting all students with the instructional materials rather than leaving it as an option for only those interested, there is an ideal approach to presenting the materials in the classroom. If the class meets in several one hour periods each week, one such period can be given over to analytical and computer training for seven or eight weeks. The first session, then, might be devoted to a general discussion of the computer's basic features and capacities, the forms of input and output, the process of data manipulation within the computer, the arrangement of data in data sets, and the preparation of source program (e.g., DAFPACK) cards. In this session, students can be given the self-instructing document AN INTRODUCTION TO DAFPACK to guide them through their first attempts at programming. It would also be necessary to arrange an adjunct session to instruct the students in the use of the keypunch.

At the second session, to which the students are expected to bring their output, discussion can be directed to reading and interpreting the tables and statistics provided by the DAFPACK program. Since the "canned" problem in the instructions makes use of chi-square as a test of significance, some attention can be given in an elementary way to this. At some point prior to or after the discussion, students can be given the two supplementary documents in the instructional materials, CHI-SQUARE and ON CONSTRUCTING AND READING TABLES. As described earlier, both of these present an elementary discussion and review of the concepts considered in class in conjunction with the output obtained from the exercise.

Because the instructional documents are cumulative in their development, the relatively simple replicated relationship becomes more complex as additional questions are put to the data. This characteristic of the instructional sequence provides for an easy shift to additional topics in the third and fourth sessions. These new topics include controlling for given variables, interpreting multivariate tables, and examining frequency distributions.

By the fifth or sixth session, students would have not only some conceptual grasp of sociology from their non-computer work but also enough facility with programming and the logic of approaching data that they begin to ask their own questions. At this point, meaningful discussion can be devoted to the derivation of operational questions from theoretical statements. In this respect, the remaining instructional document, THE RELATION OF THEORY TO SOCIAL RESEARCH, provides a series of exercises to guide the students' first efforts. Whatever additional sessions are needed are given over to the discussion of these exercises. Finally, at what would be approximately the mid-point in the course, those students who want to pursue computer based projects can be given descriptions of the available data sets and the codebooks they need while the remaining students pursue their interests in some other kind of project.

In our first experience in using these instructional materials, there were significant departures from the ideal procedure. In large measure, these departures are attributable to the fact that the completed materials were not available until nearly the beginning of the academic year. Thus, the instructors had little time to incorporate the materials into their respective course designs. Consequently, each of the three introductory classes had a different experience. (4) One instructor required all students in his class to do many of the exercises in the curricular materials although the students were not required to use the computer for their term projects. Two other instructors made the computer exercises and an empirical paper an option for interested students. In neither of these classes were lectures devoted to the computer and its use in our program; students were given the largely self-instructing materials and offered assistance as they needed it.

As noted at the beginning of the paper, we are unable to report the results of any carefully controlled evaluation. To be sure, many of the hundred or so students who were in the introductory classes never approached the computer, and many others only did the first few exercises required by the one instructor. Nevertheless, the most basic reports of computer use are very encouraging in view of the largely voluntary nature of student participation. About forty-five students did use the computer to varying degrees. Some made only a few runs while others made fifteen or more. On the whole, those who used the computer submitted an average of nine runs each with an average total machine time of thirty-six minutes for each student.

Applications in Other Courses

As the foregoing suggests, the thrust of our program was directed at the introductory sociology course. Early in the project, however, it was apparent that benefits could be derived from the use of these materials not only in the beginning course but in other sociology courses as well. For example, students with these analytical skills would be clearly free to pursue their empirical and substantive interests in upper level courses since there is little doubt that instructors would encourage and welcome empirical research papers. Moreover, in looking at the contents of the data sets that have been created, it is also clear that such possibilities exist for most of the upper level sociology courses currently being offered. In all fairness, however, we should state that these possibilities remain largely unexplored at the present time. Nevertheless, in one course--research methods--many of the materials have already been used extensively. While it would be inappropriate to present a detailed discussion of the structure and content of that particular course, brief mention of the manner in which the materials were employed will serve to make the possibilities for wider application more concrete.

As background, it is important to note that none of the methods students were acquainted with DAFPACK and none had any experience with sociological research techniques aside from their varying degrees of familiarity with the sociological literature. Moreover, since the course is offered in only one term and typically attempts to acquaint students with a variety of major research methods and analysis techniques as well as providing some introduction to statistics, little emphasis was given to the derivation and computation of statistics. The focus, rather, was on the correct use of statistics in given situations and on their interpretation. It is from this perspective that the availability of the data sets in particular proved to be of such great value.

In an early session, the use of DAFPACK was explained and the students were then asked to obtain frequency distributions for the measures of social, economic, and demographic development contained in the International Development data set. After giving some orientation to the structure and content of the output which was obtained from this exercise, the data then formed the basis for an examination of the concepts of central tendency and dispersion. Indeed, this particular data set had the advantage of consisting mainly of interval level variables so that a meaningful discussion of mean values could occur. On the other hand, there was sufficient skewness along many variables so that comparisons between the median and the mean as measures of central tendency could be made easily. Moreover, the statistical and substantive inappropriateness of using either of these measures for a nominal level variable such as the continental and regional location of any given nation was readily apparent. In general, then, and perhaps most important, the students were being introduced to the computer at the same time that they were having meaningful exposure both to reading output and to interpreting summary statistical measures.

In further sessions, we moved from the description of univariate distributions to bivariate and multivariate tabular relationships. This type of experience was structured in order to introduce the students to a new data set and its associated codebook and thereby to a greater range of variable types. They also gained greater facility and confidence in the use of the computer and acquired skills at a more sophisticated level of analysis. Again, the output itself was used to facilitate familiarization with table reading and interpretation as well as to explain the use and meaning of certain measures of association (e.g., contingency coefficients, tau, etc.) which appeared on the output.

For the purposes of the exercises outlined above, the students were given handouts reproducing the card set-ups in addition to the unrevised version of the DAFPACK documentation. For the following series of exercises on correlation and regression, they were asked to obtain a correlation and regression matrix without benefit of any guidance in setting up the run. In this instance, we again used the International Development data set given its appropriateness for this level of analysis, and it quickly became apparent to the students that, compared with their first run, all that was required was the simple substitution of one program designation card in order to obtain the needed output. Once they had the output, the differences between correlation and regression and the application, interpretation, and utility of these measures were then considered. In short, not only were the specifics of another program learned but the ease and flexibility of the computer package itself was also made evident.

By approximately mid-term, the use of the computer was essentially non-problematic and the learning of new analysis routines came readily. The computer, in effect, became a tool for research. Thus, incrementally speaking, in approaching additional topics such as index construction, scaling techniques, and tests of significance, much less emphasis was placed on the mechanics of utilizing the computer while more emphasis was given to the theory, utility, and interpretation of such methods.

One additional point can be made about the use of the data sets in the methods course. Since a research paper, preferably based on secondary data analysis, was a course requirement, students were urged to select a research problem early in the term. By so doing, all of the techniques and experiences acquired as the course proceeded could then be applied to their own research problems. For example, after the in-class discussions of describing univariate distributions, the students were encouraged to examine the frequency distributions of their own variables. Not only did this allow a student to apply his computer skills, but it also allowed an early and progressive approach to a substantive topic of his own interest.

There is little doubt that the availability of these data sets and associated codebooks was critical to the goals of the course. The students were able to familiarize themselves with a variety of types of survey and archival data and to acquire great facility in their use. That their range of interests was well served is indicated by the fact that the 13 students in the course used seven of our eight data sets for their research papers. Finally, while some of the students expressed certain reservations about the ease of transferring DAFPACK skills to other computer programs, their course evaluations generally indicated a favorable attitude toward this aspect in the course.

Discussion

We have presented in this paper the programmatic features of our effort to bring quantitative and analytical methods utilizing the computer into the sociology curriculum. It would be appropriate at this point to give brief consideration to some advantages of our approach and to mention some of the problems which we have encountered.

Let us re-state that the central problem from the beginning has been to strengthen the development of quantitative and analytical skills in the introductory course within the framework of our educational goals and the limitations of our material resources. Given this, a number of advantages are readily apparent. First, there is no requirement for a huge computer installation or for extensive on-line time-sharing facilities. Although our program has been developed using Oberlin's computer, we feel that the basic approach is suitable for other small liberal arts colleges with somewhat more limited facilities. To complement this advantage, there is no requirement that students bring any knowledge of the computer or of programming into the first course since we utilize an uncomplicated canned program package. There are several sources from which these packages may be purchased or one may be developed locally as we did.

Another advantage is offered in the flexibility of our approach. On the one hand, as a teaching resource our materials impose no constraints on us such that we cannot revise one or several parts without scrapping the entire set of materials or completely changing our approach. On the contrary, any of the materials can be changed readily in response to student interest and evaluation or to departmental considerations of suitability. Indeed, the overall degree of flexibility is not so much a function of hardware and software limitations as it is of the resourcefulness and imagination of the persons preparing, assembling, and revising the various materials. On the other hand, as a student resource the availability of a wide-ranging and easily changed data set library maximizes the possibility that any particular student will find variables appropriate to his interests within one of the data sets. An additional advantage in this respect is that students can work with good quality data without the requirement that they collect them and without the disappointing and frustrating results that such first efforts often produce.

Yet a third advantage lies in the suitability of the materials--particularly the data sets--for almost the entire range of more advanced courses offered in the department. In the previous section, we illustrated the use of the materials in the course on methods of sociological research. Beyond this one course, our materials are clearly appropriate for empirically based projects in courses on the family, the sociology of religion, demography and population, political sociology, and so forth. Moreover, although most of our data sets focus on sociological research, there is no reason anthropological research cannot be represented among them and, in fact, we plan to move in this direction in the future.

One final "advantage" is reflected in the enthusiastic reception of the materials by most of the students who participated in the first experience. The format, exercises, and technical requirements of the instructional materials are so presented as to promote informal networks of peer support. Put another way, any particular student who is but one step ahead of his fellows in an exercise can serve as an instructor for others even though he has in no way mastered the DAFPACK package or the instructional materials. Indeed, we have seen a sense of comradeship develop as each student takes his "turn" or reveals his accomplishments.

A number of problems have also emerged from our experience. We will mention several and pose them as questions to be answered according to the particular goals and requirements of any department intending to use this or a similar approach. First, should this experience be required of all students in the introductory course or should it be limited only to those who express an interest? There is merit to the argument that if all students have the experience, advanced courses can be so structured as to capitalize on these skills. There is equal merit in noting that such an experience may discourage some students from taking additional courses or even the introductory course itself. This is especially so for those students who have a more "humanistic" orientation to their interests in the social sciences.

Additional questions relate to the place of the materials in the introductory course design. Should they be presented as part of the regular materials for the term and require no additional class time, or should they be treated as supplementary in a manner similar to the laboratory sessions in the physical and life sciences? Also, should the materials be presented in the first part of the term, in the middle, or toward the end? Certainly students need time to develop their facility for data manipulation if they are to do a computer-based project. At the same time, they need some opportunity to learn the basic substance of the discipline in order to have some grasp of what they are doing or intend to do.

Finally, who is to assume responsibility for maintaining and revising the program materials and especially the data sets? There is much to be gained from keeping the materials relatively current in order to meet student interests. Is this requirement best met by rotating assignments to individuals or is it more properly the concern of the entire department where each contributes to a joint effort?

To conclude, then, we intended this paper to be a preliminary report of our activities in attempting to bring methodological and analytical skills into the introductory sociology class. We have presented the core of our approach, its application to other courses, and some advantages and problems we have found in its implementation. In the absence of a more rigorous evaluation of its impact, we feel that we have not only developed a basically useful program for our purpose, but also that we have provided a valuable resource with which to provide meaningful training in undergraduate sociology.

FOOTNOTES

* This project has been supported by a National Science Foundation College Science Improvement Program grant (GY-5303) to Oberlin College. We also wish to acknowledge the contributions of our colleague James Walsh to the preparation of the materials discussed in this paper.

1. Oberlin's computer has 128K of memory with approximately 100K available for user program packages. In addition to keypunches, the computer center facilities include the following peripheral IBM equipment: a 2540 card reader/punch, a 2314 disk facility (5 drives), two 2400 Model 1 tape drives, and a 1403 Model N1 printer.

2. The program, called DAFPACK, is a FORTRAN-based general purpose package which provides various types of statistical analysis routines as well as several data transformation and copying functions for any data which can be read in as positive integers. Analysis routines include histogram, frequency distribution, cross-tabulation and chi-square, linear regression and correlation, nominal response profiles, Guttman scalogram analysis, and multivariate tables of cell means and frequencies. Each analysis routine provides a variety of statistics, although the appropriateness of any given statistical measure for the data input is left to user judgment. While not exceptionally efficient, DAFPACK permits several user options (e.g., total or partial edit or no edit of code values), it can be linked to external programs, and it can accept extended FORTRAN sub-routines for additional data processing.

   For the purpose of most runs, the user need only provide system control cards, a DAFPACK control card giving information on the data set employed and user options, a format card, an analysis routine card, cards specifying the item or items to be used in the analysis routine, and a final card which puts DAFPACK into its transformation phase. More extended use (e.g., recoding, selecting sub-samples within the data, etc.) requires the addition of other control cards specifying these operations.

3. To date, we have relied heavily on the resources available within our department for these data sets. Oberlin's membership in the Inter-University Political Consortium allows us access to other data sets, and we are also exploring the possibilities of purchasing or otherwise obtaining additional data from research institutes and from several individual researchers.

4. We should note at this point that the course was divided into two segments. Students could elect to take the course for the first half of the term only or for the entire term. As it was structured, the first half had primarily a lecture format while the second half consisted largely of topical seminars and independent study projects. With respect to our program materials, they were used in all three introductory classes in the second part of the term. The enrollment figures cited below refer to those students who elected to take the introductory course for the entire term.

CONSORTIUM COURSE: COMPUTER SCIENCES, STATISTICS AND THE SOCIAL AND MANAGERIAL SCIENCES

David Milne, Paul Benson, and George Weber
Bucknell University
Lewisburg, Pennsylvania

Description of Course. A computer-centered course has been developed at Bucknell University for students in the social and managerial sciences. The course, which has been referred to as the Consortium Course, is intended to introduce the student to elementary statistics accompanied by computer skills and field applications in the discipline in which the student plans to major. The quantitative methods are geared to those students at the freshman-sophomore level who do not have the inclination to gain mathematical preparation through a more traditional mathematical program beginning with a calculus sequence. The course has been offered once, during the 1970-71 academic year. Approximately 80 students took the course.

The course is a two-semester course, or two credits toward a 32 credit degree. In general the week involves three 1-hour lectures and a 2-hour lab. Responsibility for instruction is shared by the departments of mathematics, computer sciences and various departments in the managerial and social sciences in approximately a 1/2:1/4:1/4 ratio respectively.

The mathematical segment of the program has been divided into two semesters. A "pre-statistical" mathematics semester includes topics of set theory, logic, number systems, vectors and matrices, variables and functions, and some elementary concepts of the calculus. The second semester is devoted to what might be considered core statistical topics from probability through distribution theory to regression and analysis of variance. This division between mathematical topics propaedeutic to statistics and statistics itself had unfortunate consequences which will be discussed shortly.

Computer instruction is also divided by semesters. The first semester is devoted to general computer theory and logic and programming in Basic Language. The second semester covers programming in FORTRAN IV and the use of packaged statistical programs in correlation and regression, analysis of variance, etc. In August, 1969 an XDS Sigma-7 computer system was installed at Bucknell. In the following year 15 terminals were installed in various departments and dormitories throughout the university. The use of the computer here, outside of teaching and research, is rather minimal. Hence, our 2,600 students are in the somewhat unique and pleasant position of having almost unlimited access to a third generation computer.

The departments in the managerial and social sciences who have expressed interest in the Consortium Course are Accounting, Business Administration, Economics, Education, Political Science, Psychology and Sociology. Their major role is to provide substance and relevance to the mathematics, particularly for the non-mathematically oriented student. This may be done through the development of problems for illustrative examples, homework assignments and tests, through the assignment of appropriate readings, and through instruction and discussion in the two hour labs.

These departments share the instruction of the two hour, weekly labs with computer sciences. Some flexibility is maintained regarding the extent to which students in various majors receive training on the computer. Some departments such as Business Administration wish their students to have as much computer instruction as possible. Other departments reduce computer training by offering their own laboratories as an alternative to some of the 2-hour labs given by computer sciences.

Another role of the departments is to expand certain topics as their discipline requires. For example psychology requires more analysis of variance than the other departments. Therefore students intending to major in psychology were introduced to analysis of variance designs early in the second semester. Designs of up to 4 factors including factorial, repeated measurement, nested and hierarchical designs were introduced, initially as exercises in experimental design and cookbook computations by special handouts and computer programs. The mathematical rationale for the designs was given as the students' statistical training proceeded throughout the semester. Other departments stress regression and other linear analyses.

Rationale for the Course. One reason for the Consortium Course, naturally, was to provide computer skills to the student early in his academic career, particularly with regard to the types of analytical problems he will encounter as his major develops. Actually, the course

4 ,8

developed with a somewhat different emphasis in mind. It was an attempt to cope with the chicken-and-egg nature of introducing statistical and other quantitative and analytical topics into social science and business curricula.

Like many schools Bucknell encourages independent student study. A thirty two course program leading to a BA or BS degree was adopted in 1963. This change from a basic three credit hour course to a four credit hour course was not accompanied, necessarily, with an increase in classroom hours. The change was instituted to stimulate independent student work. Independent study courses and an honors program have been an integral part of the curriculum for some time. Last year a January Program was inaugurated for innovative student projects during the month.

Many projects undertaken by students demand some command of quantitative and methodological skills. But it has not been an uncommon observation in the past that those students who do best in methods courses are those who have actively participated and used analytical methods in projects. Which comes first,the methods courses or the projects courses? the tools or the product experience? the skills or the awareness of where and how those skills are needed?

The Consortium Course is an attempt to trace this chicken-and-egg problem at least part of the way to its embryonic answer. Statistics are to be introduced buttressed by problems, readings, and discussions pertinent to the students' major interests. An attempt is made to involve the student as early in the course as possible in original quantitative analyses, to some extent of his own choice. One of the chief means of achieving this is through the use of data banks. For example the psychology department is using or planning to use the following data banks.

1) Test, attitudinal and performance variables of Bucknell students collected by our Counseling Center.
2) Clinical and behavioral data on mental retardates from a nearby institution.
3) Observations and telematrized physiological data on baboons living in an open air, research field cage at Bucknell.
4) Developmental data on local children in Head Start or other community or Bucknell sponsored programs.

This data not only allows the student opportunity for original statistical analyses but, since it is data on local populations, the student may find opportunity to add data in the form of an original research problem. Other departments have initiated efforts to obtain data banks relevant to their own disciplines.

Problems Encountered in the Course.

Curriculum. The decision to attempt to tease out pre-statistical mathematical topics and introduce them in the first semester was made early and it was made for a variety of reasons. The primary reason was to permit a better integration of mathematical instruction with computer and major department instruction. For example the manipulation of matrices might normally appear in the second semester of a statistics course. But the postponement of this topic so long poses a problem for instruction in computer programming. The separation of elementary concepts of differentiation and anti-differentiation from continuous probability functions permits a department such as economics to develop the concepts more fully relative to their needs.

But this separation of pre-statistics from statistics led to a major problem. The departments in the managerial and social sciences were rather readily able to agree on a course outline. But when the outline was submitted to the Mathematics Department it was charged that the curriculum did not make mathematical sense. It was labeled a "supermarket" of mathematical topics. The departments countered that the program met their needs and while the inclusion of so many varied topics meant limited coverage and little mathematical continuity this type of treatment of quantitative subjects was now being carried out in various courses offered by the different departments.

The problem of mathematical training in the liberal arts curriculum as an alternative to a calculus sequence is hardly a new or unique one. An investigation of how this problem is being handled at other colleges and universities revealed a wide variety of attempted solutions. In fact there seem to be almost as many solutions proposed as there are institutions. The problem has often come under the scrutiny of individuals and professional groups. Recommendations that come forth usually demand more time and training than many departments are willing to add to their students' already over-structured programs. It is probably to be expected that the subject matter of the Consortium Course will undergo continual revision subject to the performance and reactions of the students in the Consortium Course and subsequent courses in their major.

Another problem was the fact that the course encompassed students who took the course only because of departmental requirement or encouragement. Some of them, not overly fond of mathematics in the first place, felt that the computer added only "busy work." To a great extent this was due to the recent installation of our present computer system. This was particularly true of the terminals and time-sharing facilities. Many of the faculty are just beginning to become acquainted with the system. It is clear that if the Consortium Course is to succeed maximum utilization must be made of all computer facilities.

Finally, as an aside, it might be noted that a number of the faculty important in the planning and administration of the course were in transition or on leave. This is a temporary problem but it was a very real one.

USING THE IMPRESS SYSTEM TO TEACH SOCIOLOGY

James A. Davis
Dartmouth College
Hanover, New Hampshire 03755

Background

IMPRESS (Interdisciplinary Machine Processing for Research and Education in the Social Sciences) is a largish computer system designed by Edmund D. Meyers, Jr. of Dartmouth College so that the man in the street--or more exactly, the sophomore at the teletype terminal--can obtain quick and convenient analyses of the large data files characteristic of social science.

Since the system was described by Professor Meyers at the 1970 Conference, I shall not do so here, particularly since delegates to the conference will have the opportunity to see it in action at Dartmouth. For now, let me merely sketch its main features: 1) a large, heterogeneous data bank of social surveys, census data, anthropological materials, economic time series figures, etc. is permanently stored in the memory of the computer. 2) A user goes to any convenient teletype terminal, calls the program IMPRESS*** and begins a conversation in English with the infernal machine. 3) The system asks him to choose a specific study, next to select from two to eight variables in that study, and then to map his data as he chooses (as raw numbers, dichotomies, grouped categories, normalized values or whatever). This done, the system allows him to order a variety of statistical routines ranging from simple percentage tables to fancy multiple regression analyses.

The important point is that the system is designed to be inviting: there are no cards to punch, no problem of turnaround time, no programming skills required as it is a library (canned) program, and the entire process is conducted in reasonably straightforward English questions and answers. One may think of IMPRESS as a computational analogue of Dartmouth's Baker Library--ample, comfortable, inviting, and open stack.

But IMPRESS is not a toy. It is expensive and elaborate and over the long run can only be justified if it comes to play a major role in undergraduate instruction. Dartmouth has no graduate training in social science fields aside from psychology and faculty research is not a sufficient market to support the system.

In a companion paper, Professor Meyers presents some hard data on student usage and reactions. My aim is a much more impressionistic account of the progress and problems involved in developing applications of the system.

Research Training

Let me begin by distinguishing between two applications of IMPRESS: first, to teach research methods, and second, to teach sociological content. The distinction is made because the first seems to be very easy and the second very hard.

Lest you think we are hiding something, let me repeat that research training is not necessary for meaningful work with the system. We can cite you many examples of quite innocent users who have emerged with solid understandable findings. The range goes from itinerant foundation officials to students at high schools linked to the Dartmouth system.

Nevertheless, we have come to rely on a one quarter introductory Sociology methods course, Sociology 7, as a central part of the IMPRESS project. There are two reasons here. The obvious one is that research methods involve a solid corpus of ideas and procedures that are just not self-evident. Some of this is vocabulary--the meaning of various coefficients, statistical tests, concepts of measurement, etc. Perhaps more important is training in the underlying strategy of research; in particular, the logic of multivariate analysis. The less obvious reason for the course is that we aim to diffuse a quantitative stance among the generality of our students. In my opinion, one of the major problems in undergraduate Sociology teaching is the following circular process: if students in substantive courses have not taken any methods courses, the teacher tends to avoid statistical content in the class room; therefore, the student sees no necessity for training in research methods; consequently, he tends to take methods courses sparingly and as late as possible. As a result, the students in substantive courses will not have taken any methods courses...and so it goes. We aim to break out of this circle by exposing as many of our students as we can to rudimentary methods content as early as we can.

Given this target, we took two steps. First, we simply moved our Sociological methods course down to the introductory level. At Dartmouth, a beginning student may take any one of four courses as his introductory Sociology course: a methods course, a theory construction course, a social problems course, or a "standard" survey course. Since I teach the methods course, I lack a completely judicial stance, but I think the strategy has been successful. Enrollments in the methods course are substantial (by local standards), course ratings are only mildly abusive, and most of our majors seem to complete the course by the end of the sophomore year.

The second step was to develop a special content for the course; sort of a marriage between a traditional sociological methods course and an elementary statistics course. Our "tricks" are these: 1) We teach data analysis only, paying little or no attention to problems of data collection such as questionnaire writing or interviewing or to abstract philosophies of science. This deletion enables us to cover a good bit of ground in one quarter and we actually believe that one can write better questionnaires if he knows a lot about what must be done with the results. 2) We limit the alternatives severely. Rather than teaching all the various ways data can be analyzed, we teach one single system stressing percentage tables and the Q-Gamma family of measures of association. We believe the beginning student is better off knowing how to use a limited repertoire of tools than being vaguely aware of dozens of possibilities. 3) We go very lightly on statistical inference, pretty much limiting ourselves to two procedures--confidence limits for the coefficient Q and the Chi Square test for a contingency table. Perhaps the compromise is an uneasy one, but our belief is that statistical inference is so counter-intuitive for most students that one just can't dig into it in one quarter and do anything else. 4) We practice, practice, practice. There are weekly assignments involving IMPRESS and before the quarter is over each student will have to run dozens of tabulations. Here, of course, is a major asset of IMPRESS and time-sharing. We need no labs, no cards, no calculators, no schedules. We just tell them to run IMPRESS over and over again.

The result of all of this is that by the end of the term our students can not derive the binomial or discuss the relevance of Heisenberg's principle for Sociology, but they can run data, draw reasonable conclusions, and write them up without gross technical errors.

The advantages of IMPRESS for research training and the advantages of research training for IMPRESS usage seem so obvious that I will drop this side of the application problem here.

Teaching Content

Our first goal in IMPRESS is to teach a reasonable number of students enough research methods so that they can use IMPRESS for their own research. I don't have any hard data but I think I can safely say that it is becoming "normal" for students in Sociology and Government to write data based term papers and it is not rare for teachers in these departments to assign them IMPRESS exercises as a part of substantive courses. In Sociology, such assignments are now built into several substantive courses. For example, Avery Guest in Sociology asks his students in a population course to test various hypotheses about economic development and population growth through the various cross-national IMPRESS files. Susan Schwarz of Sociology asks her students in her Political Sociology course to attack various problems using the University of Michigan Survey Research Center files and the data from Robert Lane's *Who Governs*?

Since the IMPRESS system has actually been on the air with a reasonable volume of data only this academic year, these are distinct steps forward. Nevertheless, the project has a much more audacious goal: *ultimately, we aim to see Sociology taught as a laboratory course. That is, we would like to see our students learn the basic content of a course through their own exploration of the data.* Just as the physics or biology student can test and "discover" the fundamental principles of the field through laboratory work, so the Sociology student ought to learn by investigation rather than memorization. The proposition is certainly consistent with modern theories of educational practice and with a scientific view of Sociology. If what we are teaching is Science then the major propositions must rest on empirical evidence; if so, the student should be able to test them in some non-trivial fashion.

For the last year and one half, my major effort on the IMPRESS project has been the development of curriculum materials appropriate to this philosophy. The job has been extraordinarily difficult and progress has been painfully slow. However, I think I have learned a little and recount my progress in the remainder of the paper.

American Society: Round I

As a first attempt, I announced and offered a one quarter course, American Society, during the Spring term, 1970. The stated aim was something like this, to acquaint the student with the major variables sociologists use to examine a modern nation and to familiarize him with some of the major sociological propositions about modern America." Since at that time about the only data file available on line was the 1968 Survey Research Center study of a national sample of U.S. adults, we didn't have much elbow room in picking content. Nevertheless, even with today's array of studies, I think this was a good place to start. It gives a considerable variety of content and the topics provide a reasonable compromise between the professional interest of the sociologist and the more policy oriented interests of students. (Incidentally, one thing we learned last year is that a single national survey of the type run by SRC, NORC, and the professional pollsters provides an amazing variety of content. I suspect that the 1968 voting study has been used in a dozen or more Dartmouth classes, and in my own methods course last year it provided materials for some 50 term papers with very little duplication of content.)

My general plan was to begin with a detailed examination of the "background variables" that sociologists use in national surveys--Age, Sex, Region, Size of Place, Education, Occupation, Income, Subjective Class, Race, Religion, Church Attendance, and Political Party. My aim was first of all to acquaint the students with the major "facts of life," e.g. that there are still many Negroes living in the South, that most of the American population has not gone beyond high school, that not everybody lives in large cities yet, and so forth; then to consider the inter-relations of the variables--both to reveal certain well known findings (for example, the way SES variables hang together, the major correlates of party preference), and also to set the stage for multi-variate problems. In the second half of the course, the plan was to take particular "dependent variables" such as attitudes or values and see how these social characteristics affect them. What with the Cambodian affair, we never got that far, but we did get about two thirds of the way through the e examination of the background variables.

No formal evaluation of the course took place, though we did spend some class discussions at the end on this topic. Purely on an impressionistic basis, this is what I learned.

1) It is wise to state and hold to a methods course prerequisite. For one thing, it meant that we could concentrate on substantive problems and not be forced to spend hours of methodological briefing to prepare the student for a 15 minute problem. It also had the unanticipated effect of increasing the level of class room discussion and participation. Since the students had exactly the same methods preparation, a lot of differences were evened out and they had a common vocabulary with which to talk to each other.

2) It is unwise to abandon books. As a sort of pure experiment, we proceeded with no text, no readings, and no handouts. Rather we developed a sort of two-session cycle. In session one we discussed a particular problem with the lecturer feeding in whatever background information was necessary. Then each student worked with IMPRESS to solve the problem and in the second session we discussed our results and attempted to draw conclusions. We all soon felt the absence of reading materials. Undergraduate students, even good ones, just do not have enough substantive knowledge of sociology to see questions, implications, and connections in raw data unless they can read in some depth around the problem. What happened is that without theoretical depth, we tended to drift away from substance toward hassling over statistical and technical minutiae such as the effect of collapsing categories on the size of a correlation or non-comparability of samples in different IMPRESS files.

3) There is an optimal degree of structure for laboratory problems. At one extreme, the problem should not be stated so narrowly as to become trivial. Thus, if you give as a problem, "What percentage of the population in large cities is widowed?" the student will say to you, "Why not just tell me or give me a mimeographed handout?" and his complaint is justified. IMPRESS should not be considered an almanac of social facts. At the other extreme, the class tended to flounder when the problem was "wide open". If, for example, you give as a problem, "What are the factors that affect attitudes?" students will merely cross-tabulate some attitude

question against everything they can lay their hands on. The result is the same as the narrow extreme--a ragbag of social facts. The trick is to find problems that give the student a framework but not a straitjacket. Here are some general types that seem to work:

a) So-and-so found such and such in an early study. Is the finding still true in more recent data?

b) So-and-so found a correlation between X and Y. He didn't, however, say anything about variable Z. What do you think would happen when Z is brought into the analysis? Test it and see.

c) Professor Jones argues that Z increases Y, while Professor Smith argues that S increases Y. Neither considers Z and S. If you look at both, which do you think is the more important variable? Run the data and see.

d) When you compare nations, you generally find such-and-such. Do you think a similar pattern would turn up when one compares regions within the U.S.?

e) Generally speaking, we find a correlation between X and Y. But does it appear in all sub-groups of the population? What groups do you think would show especially high or low correlations? Why?

Note that the items have certain common threads. First, they tend to sketch the framework for the tabulations without giving specific instructions. Our experience has been that the students have a hard time jumping from purely verbal discussions to tabulation plans. Generally we have to provide strong hints as to whether the issue at hand is a partial correlation, an interaction effect, a series of zero order correlations or whatever. Second, we have learned to ask the students to make a guess or prediction before they go to the teletype. All findings are obvious after you have seen the data, but the frequent gap between prediction and results provides good grist for discussion and teaches the necessity for getting out of the arm chair. Third, the problems generally don't depend on the teacher knowing the answer. You can have a good class room discussion if the 1968 study comes out the same way as the 1950 one or if it doesn't. Such open problems avoid the chemistry lab situation of fiddling with the results until you get the answer the teacher expects.

4) Such a course badly needs some over-arching themes to maintain continuity and perspective. Without them there is a tendency to skip from topic to topic, finding to finding, without any sense of cumulation--rather like a random stroll through a museum of stuffed social statistics. Easy enough said, but hard to find. The closest I have come are these: First, "Is stratification everything?" There are so many sociological findings regarding occupation, education, and income that a heavy stress on stratification is quite appropriate in such a course. Yet the search for other variables and other components in social behavior can provide a continuing theme. A second theme is "modernization". I have been tempted to call the new version "On and Off the farm in 25,000 years" but more on this in the next section.

5) My other conclusion is simply that "it does work." Despite the many flaws in the course, undergraduate students can solve empirical problems at the teletype and they report that this approach is fresh and interesting, compared with the traditional memorization and regurgitation cycle. Nothing that happened in the quarter led me to doubt the worth of the general idea, just the merits of this particular course.

American Society II

During the last nine months, I have been re-designing the course on the basis of the experience last spring. The new version is like this:

First, I have drafted a series of handouts setting up each problem intellectually and filling in background facts and figures.

Second, we have added an international comparative section to give more variety in data and to provide the "modernization" framework.

Third, we now have some readings, V. Gordon Childe's Man Makes Himself and James L. Price's Social Facts. The first gives us our big-picture theoretical framework, the second some detailed studies for substantive depth.

A brief review of the handouts may convey the flavor:

Handout 1

Introduces the student to the philosophy and approach of a Laboratory course in substance.

Handout 2

Defines the major variables and classes of variables in the course and gives the student practice in describing theories through arrow diagrams. Throughout the course we keep going back to this handout, asking how the variables discussed in each specific problem look in terms of the general scheme.

Handout 3

The student reads Childe and attempts to classify the societies in the Anthropological file, ETHNO, in terms of Childe's stages of development. This sets up the main theoretical theme of the course--how technological progress affects social organization directly and also indirectly through affecting level of economic development.

Handout 4

The student tests the Childe approach by examining correlations between levels of social development and a non-obvious dependent variable-type of kinship system.

Handout 5

Here we jump from the Human Relations area files to IMPRESS files on modern nations. The general problem is correlates of GNP. The underlying question is whether the Childean approach has anything to say about differences among contemporary societies.

Handout 6

After reading articles on population in Price, the student attempts to arrange nations according to groups implied by the theory of demographic transition. Lurking behind is the theme question of how economic development fits into all of this.

Handout 7

The student reads an article by Lipset claiming that stable democracy is correlated with economic development and then tests it using his own operational definitions of democracy. This uses the cross-national file, X-POLITY.

Handout 8

At this point we shift from a comparative to United States focus. Student begins with census data on "Industry of Employed Persons" and looks at the American occupational structure in terms of its degree of modernization.

Handout 9

We introduce the concept of SES as a cluster involving occupation, education, and income. Students test the proposition that these three variables hang together so closely that the entire cluster can be given the single name SES.

Handout 10

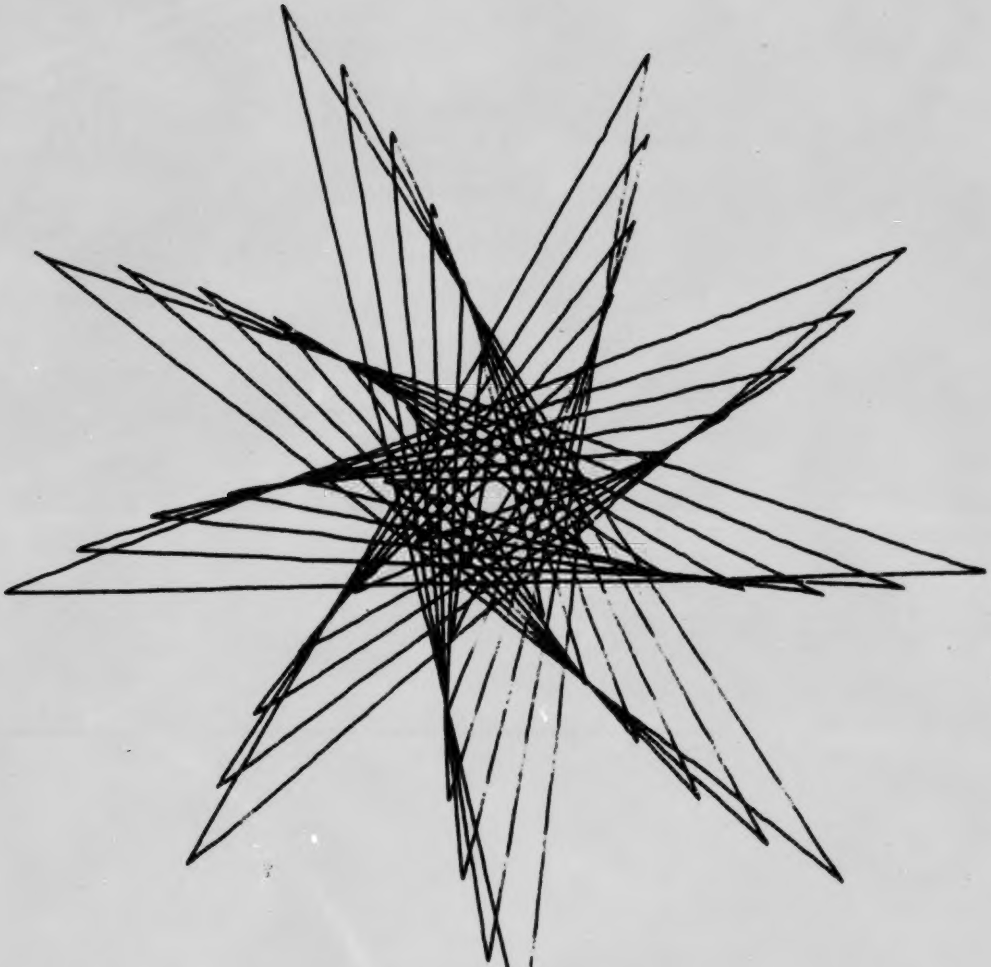
Here we examine the SES variables as a sequence or cycle by looking at father and son occupational mobility and the question of whether education is the major intervening variable. Comparative mobility data are introduced to tie back to the earlier sections and students are asked to find out whether America has more or less mobility than other nations.

Handout 11

This fourth and last handout on SES develops the concept of occupational prestige and the student is asked to predict the variables that contribute independently to subjective class placement.

Subsequent handouts, not yet completed, deal with Religion (the social characteristics of Protestants, Catholics, and Jews), the Life Cycle, Political Party Preference, generalized properties of Attitudes, Correlates of Authoritarian Attitudes, and other topics not yet firmed up.

At this point the draft ends. If the paper is accepted, a final version reporting the later handouts and the actual class room experience in Spring, 1971, will be presented at the conference.



Edward Sandifer

INTRODUCING OPTIMIZING TECHNIQUES INTO AN ENGINEERING DESIGN COURSE

D. H. Offner
University of Illinois at Urbana-Champaign
Urbana, Illinois

BACKGROUND

Within the last decade published material on optimization techniques has increased in number and sophistication to such a degree that educators who are considering introducing such techniques into courses may well be bewildered as to where to start and what to include. Even the language associated with optimization methods has become sufficiently specialized as to discourage the novice. In the preface of a publication (1) of a recent optimization symposium the reader learns that "...the conference...was...organized to facilitate... interaction between the 'hill climbers' and the 'mathematical programmers'." I found that if I were to introduce optimization methods into a course in engineering design, I would be classified among the hill climbers because of the nature of the problems I deal with in machine design. About two years ago I began to seriously consider how I might incorporate some of the optimization methods relevant to engineering design into a senior required course in machine design. I am still in the process of developing material for effectively presenting the basic concepts within the limitations of an established course. My paper is thus in the form of a progress report showing my approach, use of computers, and some summarizing observations based on one trial run.

GENERAL APPROACH

I first decided to introduce formal optimization methods into the course in a natural manner by centering it around one of the systems already studied rather than simply adding on a series of general lectures covering the "landscape". By tying the material to a specific familiar system, it provided a convenient way to build upon existing course material as well as providing an excellent basis upon which students could study assigned material from readable references on the subject. The principal limitation was that of time. The course consists of 43 two-hour periods meeting three times per week. Thirty periods are devoted to lectures covering the analysis and design of gears, shafts, bearing and brake-clutch systems. For the remaining 13 periods students are assigned a design project involving the use of mechanical components discussed in this and the previous design courses. After careful appraisal of lecture time devoted to each component and possible time that could legitimately be taken from the project periods, two lecture periods (four hours) and four hours allotted to the project for a total of eight class hours was selected for presenting, and obtaining practice in, optimizing methods related to "hill climbing" approaches. Since over 50 percent of the mathematical models developed during theory lectures in this course have been programmed (2) for use by the instructor and students, time formally devoted to manual calculations was available for extending course coverage in this way. It sometimes happens when changing course material and format that the modifications introduced may provide unexpected assistance to subsequent course evolvement. Such was the case here. One of the computer programs developed to facilitate analysis provided an effective bridge between concepts of optimization normally taught in conventional design courses and those optimizing techniques identified with "hill climbing" methods. The manner in which the above approach is being implemented with the use of a computer can best be discussed in the context of the system and problem selected for illustrating some optimizing techniques.

SYSTEM SELECTED:

A shoe-type brake system was selected as the one around which to introduce optimizing techniques used in engineering design. One such system is shown schematically in Fig. 1. It consists of two brake shoes (links 3 and 4) pressing against drum, 6, and anchored at pins E and F. The shoes are connected to an activating device such as a solenoid represented by force F_A at pin A, by means of links 1 and 2 and connecting pins B,C, and D. This system was chosen for several reasons. The variables to be used in the problem (the 12 x- and y- coordinates of the six connecting pins) all have the same easy-to-visualize dimension (length). The student were familiar with the underlying theory (3) that related such design parameters as coefficient of friction, maximum interface pressure at brake lining, and angle of brake contact angle with the torque capacity, resultant normal and frictional forces of each shoe. The best location for the contact angle of each shoe relative to their anchoring pins (E and F) for maximum torque capacity has been established through the well known procedure taught in elementary calculus of taking first derivatives and setting the resulting relationship equal to zero. Previous courses provided the necessary background for solving graphically the force analysis of the entire system as well as understanding how an analytical solution involving force and moment equilibrium in a prescribed x- and y- coordinate system could be obtained. A design project involving such a system provided the useful and realistic context in which to present an introduction to optimization theory. A general statement of the problem to be solved was stated in this way: For the brake system shown in Fig. 1, each of the six pins (A through F) may be relocated anywhere within a 2-in. square domain centered around their present positions. Determine the position for each pin closest to its present location that will yield the smallest possible tension force (F_{23}) in link 2 when the brake shoes are exerting a given counter-clockwise braking torque T on link 6 (the brake drum). At this time the students are not aware that they have been asked to solve an optimization problem involving a non-linear, 12-variable constrained differentiable function which is undefined for certain values of its variables. The principal aid they have is a computer program developed for analyzing this type of system. It is with this program and another one that used CSMP (Continuous System Modeling Program) that several basic optimizing techniques are introduced for solving this problem.

TECHNIQUES INTRODUCED VIA COMPUTERS:

One of the first prerequisites for handling a problem of this complexity is to get some "feel" for the effect that the variables have on the desired performance (in this case, minimizing F_{23}/T within the given constraints) and to note the limitations imposed by the physical configuration and equilibrium constraints. The development of the graphical solution (see Fig. 1) by the students provides some valuable insight in these areas. Lines connecting pins with concurrent points are seen to be paths along which pins can be moved without changing the forces. This observation can be directly related later to contours of equal "elevation" of the "topographical surfaces" that can be visualized as being associated with each pin for later finding the best "hill climbing" directions. Moving pins to concurrent points demonstrates the meaning of undefined values in mathematical models and their relationship to the corresponding real system. Arbitrarily changing one or all of the pins to another location within the required domain and comparing values in the new force polygon with the original provides an approximate, but initially useful, idea of the sensitivity of the force F_{23} and torque T to such change. The torque is related to the forces by the equation

$$T = \mu F_{N63} R_c + \mu F_{N64} R_c$$

Starting in this way the students are readily convinced that a more accurate, systematic and less laborious method of attack is needed. The computer program provides such an alternative. It basically provides an analytical instead of graphical solution by determining the x- and y-component, the total resultant force and its direction (indicated by its angular position) at each given pin location. A typical output from the program is shown in Fig. 2 for a given input value of torque. The input format to be filled out by the student is shown in Fig. 3. The program contains 25 options for handling variations of the basic brake system configuration and input requirements. The option used for this problem was No. 9 with which, for a given torque, a particular pin location can be varied systematically over a given domain by specifying (on card 5 of input format) the desired x- and y- increments and number of increment changes. In this way the effect of changing two variables (x- and y- coordinates of a given pin location) at a time can be systematically investigated. The class (usually 12 to 15 students) by working together in six teams can each study one region. Each team can thus visualize their region as being associated with a general "terrain" containing their pin location and for which the yet undefined explicit function, F_{23}/T provides "elevation data" by means of the computer output. Identifying each point "surveyed" in the region with its corresponding "elevation data", students develop a simple two-dimensional "plot" of their region. Figure 4a shows such a plot with only "elevation" values of F_{23}/T at the edge points shown. The possible advantages and disadvantages of this exhaustive search technique can now be discussed in the light of their effort, which can be considerable if they chose rather small increments. They realize that, while they have narrowed the problem down considerably, there is a long way to go considering that the data for each region was obtained by neglecting the effect of change occurring in the other five regions. This activity illustrates one kind of "univariate search" in which one variable at a time is varied until no further improvement is found. However, it was soon apparent that such an approach for this problem quickly becomes unmanageable (in numbers of calculations, plots and interpretations) for twelve variables in six regions to be considered one at a time. The students are ready to learn of a more efficient method. They are asked to obtain a three-dimensional sketch of each of their

regions (see Fig. 4b) and from this "piece of terrain" they get an idea of the degree of uniformity (linearity) of the surface. From Fig. 4a data for laying out "contours" of equal elevation (Fig. 4c) can be obtained. From this "map" each team can determine the location of the lowest contour (minimum value of F_{23}/T) within their region and thus the shortest distance their particular pin can be moved. The problem would be solved for each team if the conditions were such that only one pin (theirs) could be relocated. But how about the given case where all six pins may be relocated? How far off from the correct answer would it be if each team relocated their pin in the best position at the point on the lowest contour that is closest to its present location? At this stage an introduction is given of the definition, significance of, and theory behind the gradient method. The mathematical model (object function) that was developed for this brake system is now presented and the required differentiation and programming requirements discussed.
Figure 5 gives the controlling function and required differentiation requirements. Letter designations for the pins A through F are indicated in this equation by the numbers 1 through 6. The computer program and equations used in its development proved very useful during the derivation and verification of this rather complicated function. Programming the gradients and corresponding increment changes associated with each pair of x- and y- coordinates for locating a pin was greatly simplified by using CSMP. By imagining the finite displacement of each pin (in the direction for decreasing the corresponding value of the function at its maximum rate) to occur at a certain velocity, say one inch per second, and taking a "look" at its new location, say at every 0.1 second, the process lends itself ideally to the capabilities of CSMP. In this way the gradients at all pin locations could be obtained simultaneously and new positions calculated for as large or small step size desired until the given boundaries were reached. The "path" taken by each pin as it "marched" the most direct route across its region could now be found. The original and final coordinates of pin B, original and final values for F_{23}/T are tabulated in Fig. 6 for a gradient step size of 0.2 inch. For verification and educational purposes, values of the gradient tangents and their corresponding angles for both steepest rate of ascent and descent, x- and y- coordinates for each pin and the current value of maximum and minimum value of F_{23}/T were printed out for each step. These values for the initial positions and plotted values of $F_{23}/T = Q_{23}$ as output by CSMP is shown in Fig. 7. It was instructive for students to measure initial gradient angles with a protractor from their graphical solution and compare them with the CSMP calculated values. Hopefully, the students became less awed by, and convinced of, the usefulness of the mathematical manipulations as well as more appreciative of computers as a design aid. With the problem now solved using CSMP, discussion of variable step size, its relationship to linearity and direct search methods of patterned moves, and other related considerations in "hill climbing" techniques can be discussed.

SUMMARY

By properly scheduling the lecture topics and design project, the four periods (eight hours) used for introducing these optimization approaches were placed together. Since some pins reached the limits of their region before others and would require additional computer runs and modifications to the CSMP programs, the initial condition was changed slightly so that the problem was considered solved when any pin first reached its constraint. Under these conditions time was adequate for developing and illustrating this new material and for obtaining class participation in the informal lecture-discussion format already established for the course. Required outside reading in helpful introductory texts (4,5) provided the needed background for the specific illustrations used in solving the given problem. The system selected proved an excellent one for getting everyone involved personally in a "piece of the action". However, the degree to which each team completed its homework assignments related to obtaining and using computer output varied considerably. This curtailed somewhat the pace and coverage of the material presented. We are still using the batch system approach for computer usage which at times prevented the type of continuity and immediate follow-through desired. The use of remote time-sharing consoles and inter-active graphic displays could be used here with great effectiveness as a teaching aid. Modifications to the existing computer programs could provide considerably more flexibility for illustrating search strategies and step-size variations. With slight modification to the derived object function and corresponding changes in the CSMP program, other forces could be readily minimized. The brake system selected is typical of a variety of systems to which undergraduate students in mechanical engineering are exposed, several of which could be realistic candidates for similar treatment. From published literature in the machine design field, it is apparent that little attention has been given to developing performance funtions for optimizing purposes. Two reasons for this deficiency are obvious: the difficulty of developing the required complex function and subsequent effort required in differentiation is certainly one deterrent; another one is the programming requirements for effective use of the computer in optimizing the function once the required equations are available.

I have proven to my own satisfaction that these two obstacles to optimizing engineering systems for which continuous functions are derivable need not be unsurmountable. Furthermore, through the effective use of computers, the effort is very worthwhile from an educational viewpoint and at the undergraduate level. I believe that it is incumbent upon engineering educators to provide every engineer with some meaningful and relevant exposure to formal optimizing methods. I hope this report of my current activities in this area will prove useful to other educators in engineering and other fields in their present or planned involvement with computers for teaching optimization procedures.

REFERENCES

1. Fletcher, R. (Editor), Optimization, Academic Press, New York, 1969, p. vii.
2. Offner, D. H., "Computer Programs in a Machine Design Course", Engineering Education, Vol. 61, No. 4, Jan. 1971, pp. 373-376.
3. Offner, D. H., "Generalizing the Analysis of Shoe-Type Brake-Clutch Systems", Journal of Engineering for Industry, Vol. 91B, No. 3, August 1969, pp. 694-700.
4. Wilde, D.J., Optimum Seeking Methods, Prentice-Hall, New Jersey, 1964.
5. Mischke, An Introduction to Computer-Aided Design, Prentice-Hall, New Jersey, 1968.

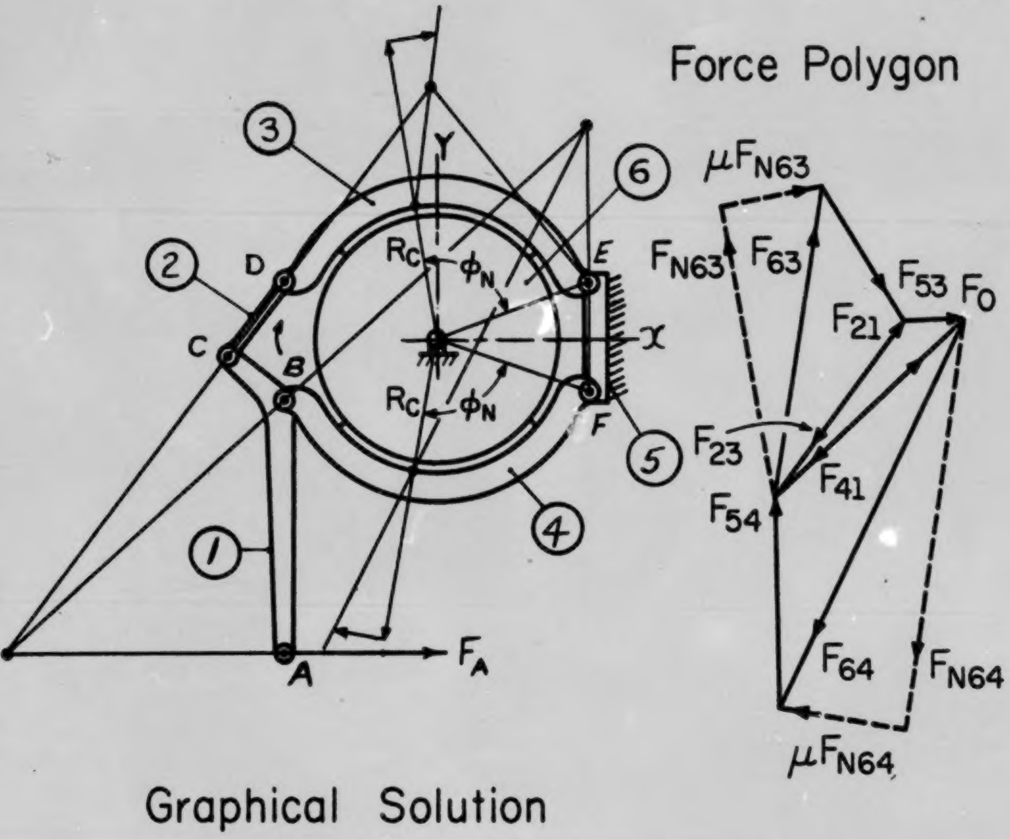


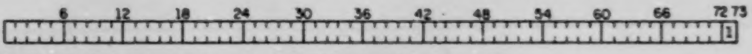
Figure 1. Brake system used to illustrate optimising techniques

```
 EXTERIOR C.W.
 FR01=  29.05 LBS    FXC1=  29.05 LBS    FY01=    0.C  LBS       BET=   C.0
 FR23= 105.95        FX23= -63.12        FY23= -85.10        ANGR23=-126.56
 FR14= 125.45        FX14=  92.17        FY14=  85.10        ANGR14=  42.72
 FR53=  63.87        FX53=  30.45        FY53= -56.15        ANGR53= -61.53
 FR54= 111.30        FX54=  -6.86        FY54= 111.09        ANGR54=  93.53
 FR63= 144.98        FX63=  32.67        FY63= 141.25        ANGR63=  76.98
 FR64= 213.94        FX64= -85.31        FY64=-196.19        ANGR64=-113.50
 FRS6=  76.09        FXS6=  52.64        FYS6=  54.95        ANGRS6=  46.23
                P3=  13.51 PSI    P4=  19.93 PSI    TORQCL=1C00.CC LBS-IN
   TORQ3= 403.93 LBS-IN    TCRQ4= 596.C7 LBS-IN    T3/T4=  0.68
 EXTERIOR C.C.W.
 FR01=  28.92 LBS    FX01=  28.92 LES    FY01=    0.0  LBS       BET=   C.0
 FR23= 105.48        FX23= -62.83        FY23= -84.72        ANGR23=-126.56
 FR14= 124.89        FX14=  91.76        FY14=  84.72        ANGR14=  42.72
 FR53= 188.76        FX53= 149.68        FY53=-115.00        ANGR53= -37.54
 FR54= 134.37        FX54=-123.56        FY54=  52.79        ANGR54= 156.87
 FR63= 217.78        FX63= -86.85        FY63= 199.72        ANGR63= 113.50
 FR64= 141.14        FX64=  31.81        FY64=-137.51        ANGR64= -76.98
 FRS6=  83.06        FXS6=  55.04        FYS6= -62.21        ANGRS6= -48.50
                P3=  20.29 PSI    P4=  13.15 PSI    TORQCL=1000.0C LBS-IN
   TORQ3= 606.78 LBS-IN    TORQ4= 393.23 LBS-IN    T3/T4=  1.54
```

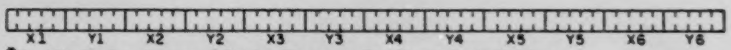
Figure 2. Typical output from brake analysis program

Engineer | Project No. | Date

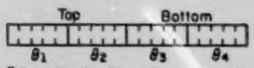
Card Data Format for Subroutine BRUTCH (SR2)
Precede Cards with Main Program Card Having 2 in Column 62
Each Problem Set Requires Cards No.1. thru No. 6. A *Blank* Card *Must* Follow *Last* Card of Problem *Sets*.


6
12
18
24
30
36
42
48
54
60
66
72
73
1

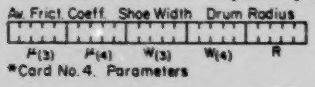
Card No. 1. Title Card (Column 1—72) Must Have 1 in Column 73


X1
Y1
X2
Y2
X3
Y3
X4
Y4
X5
Y5
X6
Y6

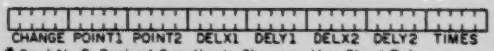
*Card No. 2. Coordinates of Pairing Elements. See Sketch for Definition.


Top
Bottom
θ1
θ2
θ3
θ4

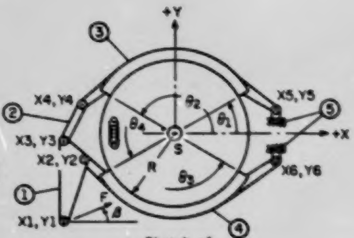
*Card No. 3. Shoe Contact Angles (Degrees)


Av. Frict. Coeff.
Shoe Width
Drum Radius
μ(3)
μ(4)
W(3)
W(4)
R

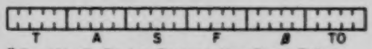
*Card No. 4. Parameters


CHANGE
POINT1
POINT2
DELX1
DELY1
DELX2
DELY2
TIMES

*Card No. 5. Desired Coordinate Changes. Use Chart Below for Instructions


+Y
③
②
①
④
⑤
X4, Y4
X3, Y3
X2, Y2
X1, Y1
X5, Y5
X6, Y6
θ1
θ2
θ3
θ4
S
R
F
β
+X

Sketch of
Brake-Clutch System



*Card No. 6. Design Options. Use Chart Below.

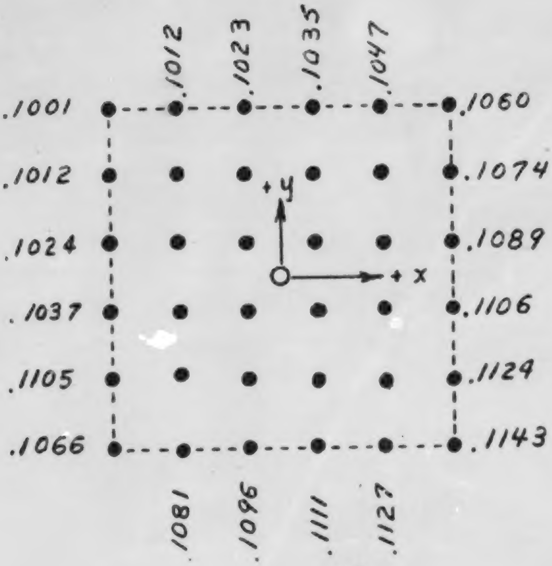
		Input Conditions										Card Data													
		System		Problem			Shoes		Changes			No. 5								No. 6					
																				T	A	S	F	β	TO
	N.	Lever	"Spring"	G: Torque F: F at 15°	G: Force F: Torque	G: T, β F: Force	External	Internal	0	1	2	CHANGE	POINT 1	POINT 2	DELX 1	DELY 1	DELX 2	DELY 2	TIMES	TORQRQ	APPLY	SOLUT	ALOAD	ANGBET	Type Option
Available Options	1.	●		●			●	●	●											T	1.				1.
	2.	●			●		●		●												1.	-1.	F	β	2.
	3.	●			●		●			●		1.	n	7.	△	▲			X		1.	-1.	F	β	3.
	4.	●			●		●				●	1.	n	m	△	▲	▽	▼	X		1.	-1.	F	β	4.
	5.	●			●			●	●												1.	1.	F	β	5.
	6.	●			●			●		●		1.	n	7.	△	▲			X		1.	1.	F	β	6.
	7.	●			●			●			●	1.	n	m	△	▲	▽	▼	X		1.	1.	F	β	7.
	8.	●				●	●		●											T	1.	-1.		β	8.
	9.	●				●	●			●		1.	n	7.	△	▲			X	T	1.	-1.		β	9.
	10.	●				●	●				●	1.	n	m	△	▲	▽	▼	X	T	1.	-1.		β	10.
	11.	●				●		●	●											T	1.	1.		β	11.
	12.	●				●		●		●		1.	n	7.	△	▲			X	T	1.	1.		β	12.
	13.	●				●		●			●	1.	n	m	△	▲	▽	▼	X	T	1.	1.		β	13.
	14.		●			●	◗	●	●											T	-1.◗				14. ◗20.
	15.		●			●	◗	●		●		1.	n	7.	△	▲			X	T	-1.◗				15. ◗21.
	16.		●			●	◗	●			●	1.	n	m	△	▲	▽	▼	X	T	-1.◗				16. ◗22.
	17.		●		●		◗	●	●												-1.◗	1.	F		17. ◗23.
	18.		●		●		◗	●		●		1.	n	7.	△	▲			X		-1.◗	1.	F		18. ◗24.
	19.		●		●		◗	●			●	1.	n	m	△	▲	▽	▼	X		-1.◗	1.	F		19. ◗25.

Chart Code

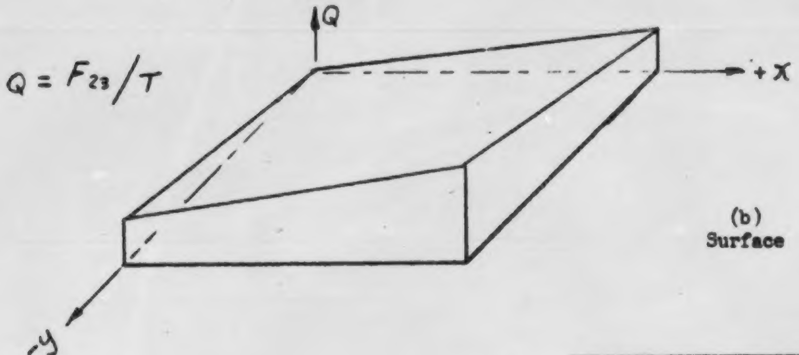
●	Desired Input Conditions
n	First Pivot Location
m	Second Pivot Location
△	x-change in First Pivot Location
▲	y-change in First Pivot Location
▽	x-change in Second Pivot Location
▼	y-change in Second Pivot Location
X	number of changes desired
T	Input torque (in.-lb) Program solves cw and ccw rotation
F	Applied Force
β	Angular Direction (degrees) of Applied Force
◗	External "Spring" System, A = -2. otherwise same values as internal. "Spring" System has equal vertical (F) forces applied at pivots X2, X4. (X1 = Y1 = X3 = Y3 = 0)
Empty ☐	= 0

* Field is Fixed Point (F6.2) unless overruled by Programmer. All dimensions in inches.

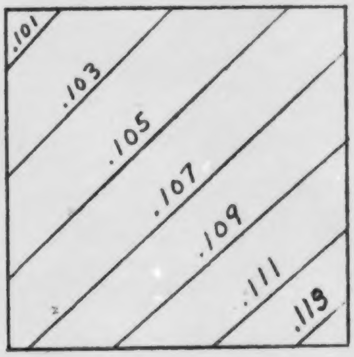
Figure 3. Input format and options for brake analysis program


.1012
.1023
.1035
.1047
.1001
.1060
.1012
.1074
+y
.1024
.1089
+x
.1037
.1106
.1105
.1124
.1066
.1143
.1081
.1096
.1111
.1127

(a)
Elevation Points



(b)
Surface


.101
.103
.105
.107
.109
.111
.113

(c)
Contours

Figure 4. Plots obtained from computer output for region, B

Object function:

$$Q_{23} = F_{23}/T = A(B_1/C + B_2/D)^{-1}$$

where $A = (y_2 - y_1)[(x_4 - x_3)^2 + (y_4 - y_3)^2]^{0.5}$

$$B_1 = (x_4 - x_3)(y_5 - y_4)(y_2 - y_1) + (x_4 - x_5)(y_2 - y_1)(y_4 - y_3)$$

$$B_2 = (x_4 - x_3)(y_3 - y_1)(y_2 - y_6) + (x_2 - x_3)(y_4 - y_3)(y_2 - y_6)$$
$$+ (y_4 - y_3)(x_6 - x_2)(y_2 - y_1)$$

$$C = [-C_1(x_5^2 + y_5^2) - C_2(x_5^2 - y_5^2) \pm C_3 y_5 x_5 \mp x_5]/x_5$$

$$D = [C_4(x_6^2 + y_6^2) + C_5(x_6^2 - y_6^2) \mp C_6 y_6 x_6 \mp x_6]/x_6$$

Note:

(a) Constants $C_1 \cdots C_6$ relate brake shoe and drum specifications;

(b) function is independent of x_1 since activating force F_A (Fig. 1) is assumed to remain horizontal in direction; and

(c) choice of ambiguous signs dependent on direction of drum rotation.

Required differentiation

$$\partial Q_{23}/\partial\phi = [B_1/C + B_2/D]^{-1}\,\partial A/\partial\phi - A[B_1/C + B_2/D]^{-2}$$
$$\cdot\,[(\partial B_1/\partial\phi)/C - (B_1)(\partial C/\partial\phi)/C^2 + (\partial B_2/\partial\phi)/D$$
$$- (\partial D/\partial\phi)\,B_2/D^2]$$

where

$$\phi = x_1, y_1, \cdots x_6, y_6$$

Figure 5 Object Function and Differentiation Requirements

Pin B step size = 0.2"					
Coordinates				$Q_{23} = F_{23}/T$	
Original		Final		Original	Final
x	y	x	y		
-10.0	-4.0	-11.0	-3.34	0.1063	0.0758

Figure 6. Original and Final Data for Region B

```
Q23MX =  1.0629F-01      Y1   = -2.0000E 01     GRD1MX=  0.0  ∞        A1MXD =  9.0000E 01
X11   = -1.COCCE C1      Y11  = -2.0000E 01     GRD1MN=  0.0  ∞        A1MND = -9.0000E 01
X2    = -1.CCCOE 01      Y2   = -4.000CE OC     GRD2MX= -1.C893E 00    A2MXD = -4.7448E 01
X22   = -1.COCOE 01      Y22  = -4.0000F 00     GRD2MN= -1.0893E 00    A2MND =  1.3255E 02
X3    = -1.3560E 01      Y3   = -1.30CCE CC     GRD3MX= -7.4166E-01    A3MXD =  1.4344E 02
X33   = -1.3560F 01      Y33  = -1.3000E CC     GRD3MN= -7.4166E-01    A3MND = -3.6563E 01
X4    = -1.COCOE C1      Y4   =  3.5000E 00     GRD4MX= -7.4166E-01    A4MXD = -3.6563E 01
X44   = -1.COCOE 01      Y44  =  3.500CE OC     GRD4MN= -7.4166E-01    A4MND =  1.4344E 02
X5    =  1.CCCCE 01      Y5   =  3.5000E 00     GRD5MX=  1.1528E 00    A5MXD =  4.9060E 01
X55   =  1.CCCOE C1      Y55  =  3.500CE OC     GRD5MN=  1.1528E 00    A5MND = -1.3094E 02
X6    =  1.00C0E 01      Y6   = -3.50CCE CC     GRD6MX=  6.1853E-02    A6MXD =  3.5394E 00
X66   =  1.CCCCE C1      Y66  = -3.5000E 00     GRD6MN=  6.1853E-02    A6MND = -1.7646E 02
Q23MN =  1.0629E-C1
```

```
                                  MINIMUM                    Q23MN  VERSLS TIME                 MAXIMUM
                                4.6618E-02                                                    1.0629E-01
   TIME            Q23MN           I                                                               I
   0.0           1.0629E-01        --------------------------------------------------+
   2.0C00E-01    9.88C8F-02        -------------------------------------------+
   4.0000F-01    9.2597E-02        --------------------------------------+
   6.0CC0E-01    8.7370E-02        ----------------------------------+
   8.0000E-01    8.2922E-C2        ------------------------------+
   1.0C00F 00    7.9104E-02        ---------------------------+
   1.2CCCE 0C    7.58C4E-02        ------------------------+
   1.40C0F 00    7.2936E-02        ----------------------+
   1.6C00E 0C    7.0432E-02        -------------------+
   1.8C0CF 00    6.8240E-C2        ------------------+
   2.0000E 00    6.6281E-02        ----------------+
   2.200CF 00    6.4347E-02        --------------+
   2.400CE 00    6.24C7E-02        -------------+
   2.6000F 00    6.0463E-02        -----------+
   2.8CCCE 00    5.8512E-02        ---------+
   3.0000F 00    5.6553E-C2        --------+
   3.2000E 00    5.4587E-02        ------+
   3.4CC0F 00    5.2612F-02        -----+
   3.6000E 00    5.0626E-02        ---+
   3.8C00E 00    4.8628E-02        -+
   4.0000E 00    4.6618E-C2        +
```

Figure 7. Typical output from CSMP optimization program

AUTOMATED INTEGRATION OF SUBJECTIVE CRITERIA INTO SITE PLANNING MODELS

David Morris
University of Virginia
Charlottesville, Virginia

The influence of computer technology on the process of optimizing the location of a facility in a civil engineering site planning problem is discussed. The general formulation of an analytical model on the basis of construction costs and social value consideration is briefly presented and typical means of data display output is illustrated. An actual student project is reviewed and the role of the computer is related to the objectives of the exercise. The philosophic benefits of computerizing subjective criteria in a learning experience is critiqued and future development of the concept is mentioned.

Introduction

The advent of electronic computers and the common central batching methods of data processing have appreciably altered the philosophy of engineering education in the last decade. Now the introduction of time sharing systems with remote consoles has made the man-machine interaction even more efficient as well as more personal to the individual. Minimization of turnaround time and the availability of diagnostic messages during the program development stage are the major advantages in the newer systems, and these features are particularly helpful when a large number of users are in the learning stages of programming. More significantly, however, the direct on-line communication between the computer and the user has added a new dimension to the way problems are being formulated and developed. New methodologies are resulting in practically all fields, particularly engineering and applied science.

In engineering, synthesis is, by nature, a process of examining alternate proposals, then refining the eventual choice down to a practical optimal solution. Many traditional theoretical concepts and their associated applications, formerly presented in the name of "design time economics", are giving way to more sophisticated approaches to even more complex engineering problems.

Similarly, in other fields such as operations research, the computer has given impetus to the development of many probabilistic tools which can also be brought to bear on engineering problems heretofore treated in a purely deterministic sense.

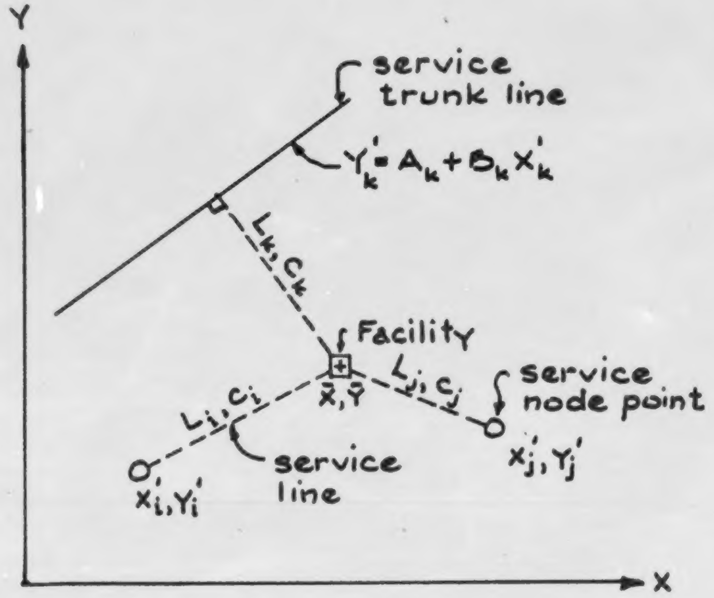
The purpose of this paper is to present a typical civil engineering undergraduate planning problem, to indicate the formulation of the mathematical model which includes both the direct engineering costs and a quantified set of associated social values, and finally, to describe the computerized input-output format used to obtain an overall optimal solution.

Engineering Planning Concepts

The situating of a facility on a given site was historically approached from primarily an artistic, rather than scientific, point of view and reasonably logical, rational norms were established for working with, rather than against, the natural setting characteristics.(1,2) Traditional engineering studies examined various locations from a standpoint of the construction costs of the facility and its services, such as power, water, sewerage, roads and communication lines.(3) Recently, the computer has provided the means to systematize cost studies and permit more quantifiable plans to be evolved. Today, on the other hand, new recognition of the social values of aesthetics, environmental controls and ecological considerations have the difficulties of precomputer age guesswork and, relatively speaking, the modeling methods have regressed. Some aid can presently be obtained from the decision making apparatus used in analyzing subjective evaluations of uncertainties and it is likely that continued application of these methods will be incorporated into general planning principles and result in more quantifiable procedures.

Optimizing Service Line Costs

Given a large ratio of site area to facility area, it becomes practical to minimize the total cost of the service lines with respect to either existing service node points or main service trunk lines. (4,5) FIGURE 1 gives the basic relationships.


Y
service trunk line
$Y'_k = A_k + B_k X'_k$
L_k, C_k
Facility
$\bar{X}, \bar{Y}$
L_j, C_j
service node point
L_i, C_i
service line
X'_i, Y'_i
X'_j, Y'_j
X

PLANE OF UNKNOWN LOCATION

FIGURE 1

Let

n = number of service lines from feeder node points
m = number of service lines from feeder lines
X_i ,Y_i = coordinates of known service node point i, such as a water tank, power plant, or adjacent service facility
X'_k ,Y'_k ,A'_k ,B'_k = variables and constants which describe a service feed line k, such as a road, power line, or river
$\bar{X}$, $\bar{Y}$ = coordinates of unknown location of facility
L_i = length of service line i
c_i = cost per unit of length of line i
F_i = cost of facility at X, Y
T_i = total cost of service line and facility

Then

$$T = F + \sum_{i=1}^{n+m} c_i L_i$$

where

$$L_i = \left[(\bar{X}-X_i)^2 + (\bar{Y}-Y_i)^2\right]^{\frac{1}{2}} \text{ for point to point}$$

or

$$L_i = [(\bar{X}-(\frac{\bar{Y}+\bar{X}/B-A}{B+1/B}))^2+(\bar{Y}-A-B(\frac{\bar{Y}+\bar{X}/B-A}{B+1/B}))^2]^{1/2} \quad \text{for point to line}$$

thus

$$T_i = F_i + \sum_{i=1}^{n} c_i[(\bar{X}-X_i')^2+(\bar{Y}-Y_i')^2]^{1/2}$$

$$+ \sum_{i=1}^{m} c_i[(\bar{X}-(\frac{\bar{Y}+\bar{X}/B-A}{B+1/B}))^2+(\bar{Y}-A-B(\frac{\bar{Y}+\bar{X}/B-A}{B+1/B}))^2]^{1/2}$$

with the unknown variables being $\bar{X}$ and $\bar{Y}$. This non-linear equation lends itself to minimization by gradient methods in which

$$\nabla T = 0$$

or

$$\frac{\partial T}{\partial X} = \frac{\partial T}{\partial Y} = 0$$

Weighted Objective Decision Theory

The field of stochastic evaluation of alternatives under degrees of uncertainty has received a great deal of attention in recent years from among the many probabilistic modeling methods, the weighted aggregate concept has particular application in the single stage event situation. (6,7) Let

u = the number of objectives in a given circumstance

O_j = objective j in a set of u

W_j = relative weight of importance to be attached to objective j

v = the number of alternate actions

A_i = Alternative i in a set of v

$P_{i,j}$ = probability of attaining objective j through alternative i

E_i = total expected value of alternative i

Thus

$$E_i = \sum_{j=1}^{u} P_{ij} W_j \qquad i = 1, 2 \ldots . v$$

or for a set of alternatives

$$[E]_v = [P]_{v,u} [W]_u$$

Once the total expected values for each possible alternate course of action have been derived, the optimal, such as minimum cost, maximum profit or maximum benefit, can be selected by inspection.

Decision Theory in Site Planning

Decision theory based on weighted objectives lends itself to modeling the social aspects in a site planning problem. It is desirable, however, to further define the concepts to be used in this application.

In common usage, an alternative implies a certain course of action which is different physically and descriptively from the other possible courses. As defined in site planning, the alternatives may be, depending on the purposes of the study, a set of various different land uses, a set of different locations for a particular facility, or a set of different type facilities. To illustrate the latter case, suppose a town is expecting to develop an industrial site and wishes to invite one of three different type manufacturers to locate there. Of major social concern are the problems of unemployment and pollution. The objectives are to minimize these aspects by selecting the proper firm. After some deliberation, it is decided that the actual employment is twice as critical as the pollution threat. This constitutes the relative weighing of the objectives. Then for each objective, one alternative is chosen which is deemed to be the most likely to achieve that objective and, likewise, one alternate which is least likely to achieve success is selected. The probable success of the other alternatives is then measured relative to these limiting cases. TABLE 1 gives the information and the resulting expected values.

Relative Weights W_j	67%	33%	E_i
Objective O_j A_i Alternatives	Employment Opportunities	Minimize Pollution	Total Expected
	P_{ij}		
Acme Electronics Co.	.20	1.00	46
W. & L. Precast Corp.	.70	.80	74
Jones Chemical Indus.	1.00	.10	70

EXPECTED VALUES

TABLE 1

In the first column under Employment Opportunities, Jones Chemical Industries is judged to offer the most potential. Hence the probability of 1.00 is assigned to it. Accordingly, the value of .20 employment probability for the Acme Electronics Company means they are judged to have only 20% of the possible maximum job opportunities. The other probabilities are determined for the matrix and the expected social values for each alternate industry are found. The W. & L. Precast corporation appears to be the optimal selection since its expected value of .70(67)+.80(33) = 74 is the maximum. Notice that the objectives have been presented in a positive sense. This is not necessary and negative probabilities could have been processed.

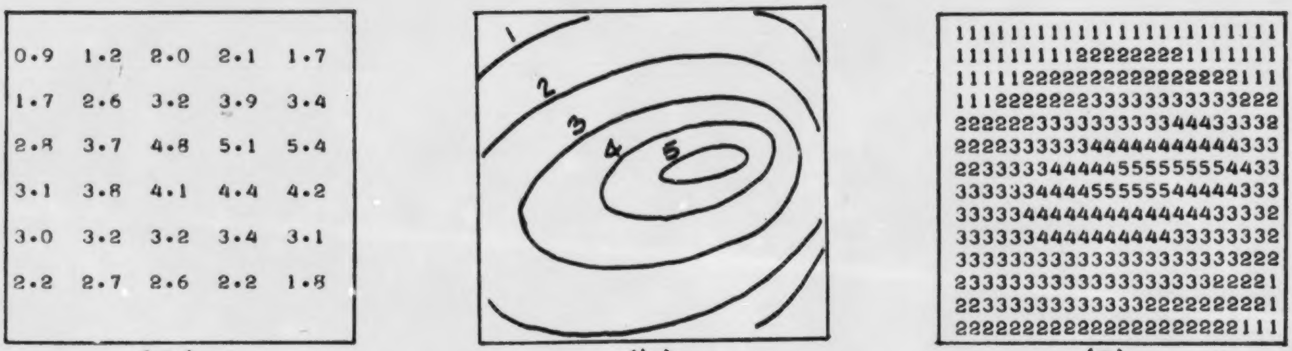
The preceding example did not directly address the problem in question, that is, the location of a predesignated type facility on a given site. However, the principles are the same when the alternate courses of action consist of determining the expected social value for each of a finite number of site locations. From such a process, it is possible to obtain an expected social value contour mapping of the site. The feasibility of the mapping is dependent on the trace on each probability function being somewhat smooth from one location to another. The charcteristic of a clearly defined map is not necessary, however, for an optimal selection from a given number of data points, nor does lack of same denote a fallacy in logic. A visibly distinguishable format is merely desirable for the purpose of interpretation.

Data Formats

Common to both the service line costs and the social value relationships is the general form of the resulting data outputs. Each can be listed in a numerical tabular form such as in Figure 2a. The figures represent the cost/benefit at a particular grid point and the optimum can be located and compared with the rest in terms of magnitude.

Another means of presentation is the graphical plot shown in FIGURE 2b. This can be done manually by making an overlay on the numerical tabular form and interpolating "contours" of some convenient interval between points. A computer plotter can also be employed.

A third method is shown in FIGURE 2c. This is the data display technique which provides a pattern effect. Integer values can be used to give direct magnitudes of values on the chart.



(a) (b) (c)

FIGURE 2

Distillation of Social Values to Cost Equivalents

The basic concepts of decision theory are predicated on relative weights and relative comparison of value. Given a situation, determine the extremum conditions and scale and probable likelihoods of each alternative accordingly. The same philosophy can be used to reduce the expected social values within a bounded region to a set of relative costs. Let

$E_{min\,(max)}$ = the minimum or maximum expected social value at a point within the site

E_i = the expected social value at point i

D = the social cost or the amount of expenditure that would be granted to locate the facility in the most desirable rather than the most undesirable position on the site, all other costs and considerations being held constant.

S_i = the social cost of locating at point i

Thus

$$S_i = D\,\frac{E_{max} - E_i}{E_{max} - E_{min}}$$

The total overall cost at location i, including both direct construction costs and equivalent social costs, is then

$$Z_i = S_i + T_i$$

and the cost contour matrix is

$$[Z]_{u,v} = [S] + [T]$$

Refining the Solution

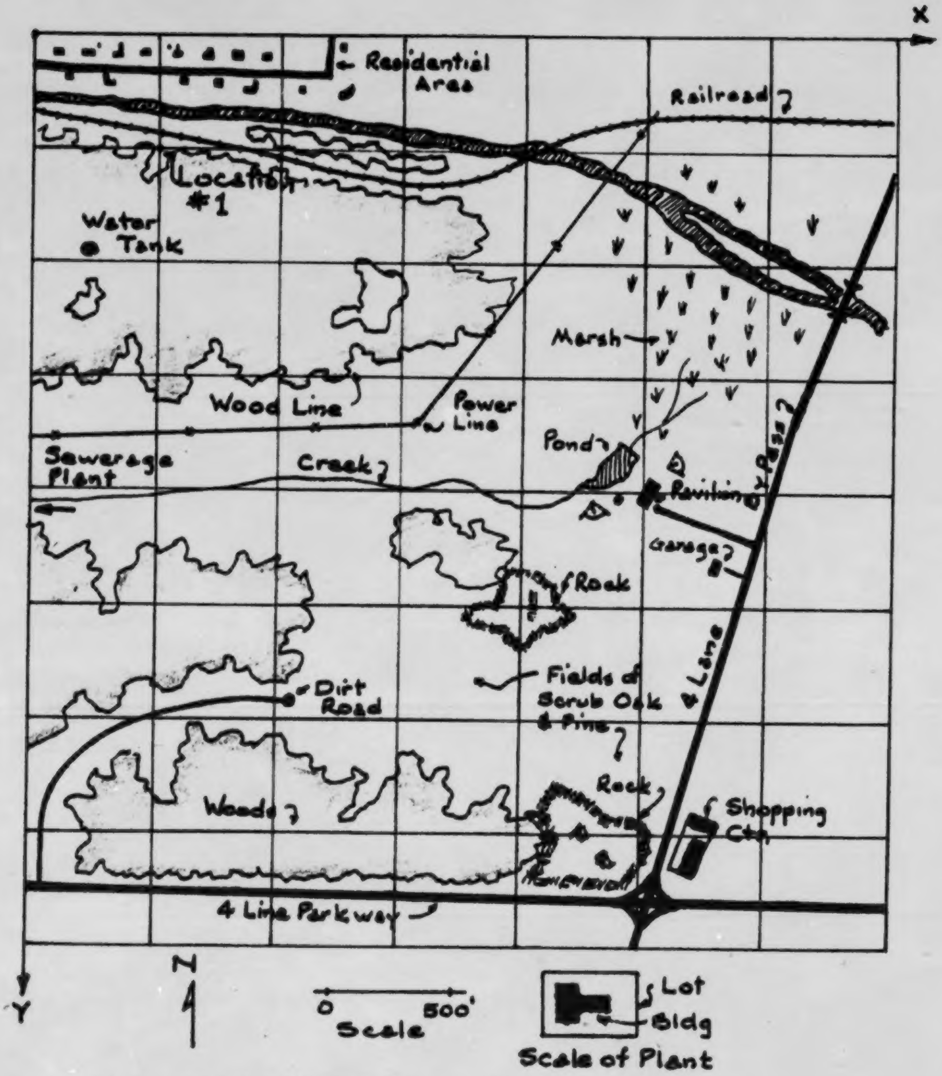
Once an initial analysis has been carried out using average costs per unit of service line length, it is often appropriate to adjust these figures to reflect the influence of certain local terrain features on the construction costs. A reanalysis is then necessary and the optimal location will change. This feedback iteration process must continue until the solutions conmerge to an acceptable difference, though fortunately, convergence is usually rapid.

The expected value analysis can also be used to make other decisions in the site planning process. These might include the optimal location of a new service node, a determination which must be made prior to the general facility location study, or, after the primary facility has been located, further examination of auxiliary facilities is often warranted. The applicability of the expected value method in refining a solution is very helpful.

Example Statement

A manufacturing firm is to construct a 3 million dollar plant to produce building components on the 640 acre tract shown in FIGURE 3. The land is across the river from an eastern seaboard city of 80,000 people. The recently constructed 4 lane bypass has changed the growth pattern of the area and the city and county officials, together with the manufacturing firm representatives, are attempting to establish a rational means by which certain social values can be incorporated into the site planning cost analysis.

Public discussion and official concern over the project has centered on three prime issues: the effect on reduction of recreational facilities, the detriment to conservation, and the aesthetic influence of the venture on future development in the area. The property itself is practically undisturbed with only a modest recreational layout, a burned-out garage and a few picnic trails occupying the space. The marsh area supports a large number of ducks and other wildlife, and a visible amount of other game enjoy refuge in the wooded areas, particularly in the region bordering the scenic parkway. A middle income neighborhood lies directly across the river to the north and the tracts south and east are similar open fields and woodland with a few single family dwellings on each. The property to the east is being cultivated except for the commercial establishments shown.


X
Residential Area
Railroad
Location #1
Water Tank
Marsh
Wood Line
Power Line
Pond
Sewerage Plant
Creek
Pavilion
By Pass
Garage
Rock
Fields of Scrub Oak & Pine
4 Lane
Dirt Road
Rock
Shopping Ctr.
Woods
4 Line Parkway
Y
N
0
500'
Scale
Lot
Bldg
Scale of Plant

THE INDUSTRIAL SITE

FIGURE 3

Other factors pertinent to the manufacturer are that the plant itself will require the following services at the estimated average costs shown:

1. Potable water @ $6/foot of pressure pipe
2. Sewerage @ $8.50/foot including manhole costs
3. Industrial water intake line @ $5.50/foot of pressure pipe
4. Power line @ $2.50/foot
5. Telephone line @ $1.75/foot
6. Access road @ $16/foot including pavement
7. Railroad spur line @ $11.50/foot

All of the above prices are based on open flat terrain without rock. Add 15% for grades over 10%, 10% for clearing timberland, and 25% for rock excavation.

It is also estimated that site work for the plant lot will increase \$3000/acre if located in timberland and the foundation costs for the plant itself will increase \$5000 if located in an area where groundwater is within 4 feet of grade elevation.

The industrial management is quite conscious of the social aspects of the problem and for the sake of public responsibility and goodwill is willing to invest up to \$25,000 towards achieving these goals. This is an amount additional to that already earmarked for either a new recreational facility or renovation of those existing.

The problem is thus to determine the best location for the plant when both direct costs and the social concerns are considered together.

Example Formulation and Solution

The organization of the problem takes the following form:

1. Establishment of social objectives
2. Ranking and weighing of the objectives
3. Selection of grid coordinate reference system for the site
4. Designation of the alternate facility locations from the grid system
5. Establishment of the probability matrix for each objective
6. Determination of the expected social value matrix
7. Determination of the social cost matrix
8. Mapping of the social cost contours
9. Establishment of the service node coordinates and the service trunk line bearings and location from the grid system
10. Formulation of the service line cost function in terms of the unknown facility coordinates
11. Obtaining the facility site cost and service line cost matrix for the site
12. Identification of the optimal location and revision of the cost function as necessary
13. Repeat of step 11
14. Addition of the social cost and the construction cost matrices, and identification of the overall optimum location
15. Adjustment of the input data based on new location after checking consistency of social values and service line costs
16. Revision if necessary

The compilation of the social cost matrix is shown in TABLE 2 and is based on relative weights of 20%, 30% and 50% being assigned, respectively, to the recreation, conservation and aesthetic objectives. The location alternative numbers follow the grid reference nodes starting with $\overline{X} = 1$ and $\overline{Y} = 1$ and proceed across and then down in the 5 by 6 grid shown in FIGURE 3.

The probability matrix is predicated on the relative value, 1 to 0, assigned to the question, "What is the likelihood of achieving objective j when the facility is located at position i?" The social values for each location are found by summing the products of P x W and the results indicate that the optimal location, E_{max}, is at location 24. The least desirable, E_{min}, occurs at location 7. These values are then used to establish the social costs from the previously derived expression for S. The value D in the equation is the maximum premium the manufacturer is willing to spend for social benefits and is the \$25,000 figure given in the problem statement. The mapping of these social costs is shown in FIGURE 4a.

The service line and site costs are found for each of the 29 reference points on the grid reference system. These results are mapped and shown in FIGURE 4b. Note how the optimum location differs from that based on social costs alone.

Finally the two cost matrices, or maps, are added together to give the total picture shown in FIGURE 4c. Further refinement can take place and the final solution to the facility location can be evolved.

Obviously, the above description omits discussion of certain salient aspects such as adjustment of costs when terrain features differ from those assumed and other practical considerations which are part of the refining process. These are judgement decisions which cannot be modeled.

Relative Weight W_j	20%	30%	50%		
Objective A_i O_j Alternative	Recreation	Conservation	Aesthetics	Social Value E_i	Social Cost S_i
	Probability of Achievement $P_{i,j}$				
Location 1	.3	.1	0	9	$24,333
Location 2	.4	.2	.1	19	21,000
.....	..	..	..	..	..
Location 7 (min.)	.2	.1	0	7	25,000
.....	..	..	..	..	..
Location 23 (max.)	.1	1	1	82	0
.....	..	..	..	..	..
Location 29	.7	.6	.6	62	6,667

TABLE 2

Computer Utilization

The above example has served as a project for 4th-year civil enineering undergraduates at the University of Virginia. The engineering school, as of September 1969, has been offering instruction in Basic language and the use of the Hewlett-Packard Digital Commputer 2000A Time-Sharing System. Sixteen remote terminals are located throughout the school. The purpose of the project, in addition to instructing the students in the principles and analytical procedures of site planning, was to enhance the students' appreciation of the full potential of time sharing systems. Specifically the objectives were:

1. To illustrate how the man-machine interaction takes place in a typical engineering problem
2. To illustrate how a typical engineering problem can be modeled in a manner which facilitates the processing of large amounts of data in an orderly, efficient manner
3. To use the computer for its basic purpose, reduction of labor, particularly that which may involve repetitive operations
4. To permit the planner (student) to reflect on his own subjective (judgement) input criteria and make adjustments in the evolving plan
5. To illustrate how the computer can be used to provide non-tabular data displays which can be directly correlated with the physical aspects of the problem.

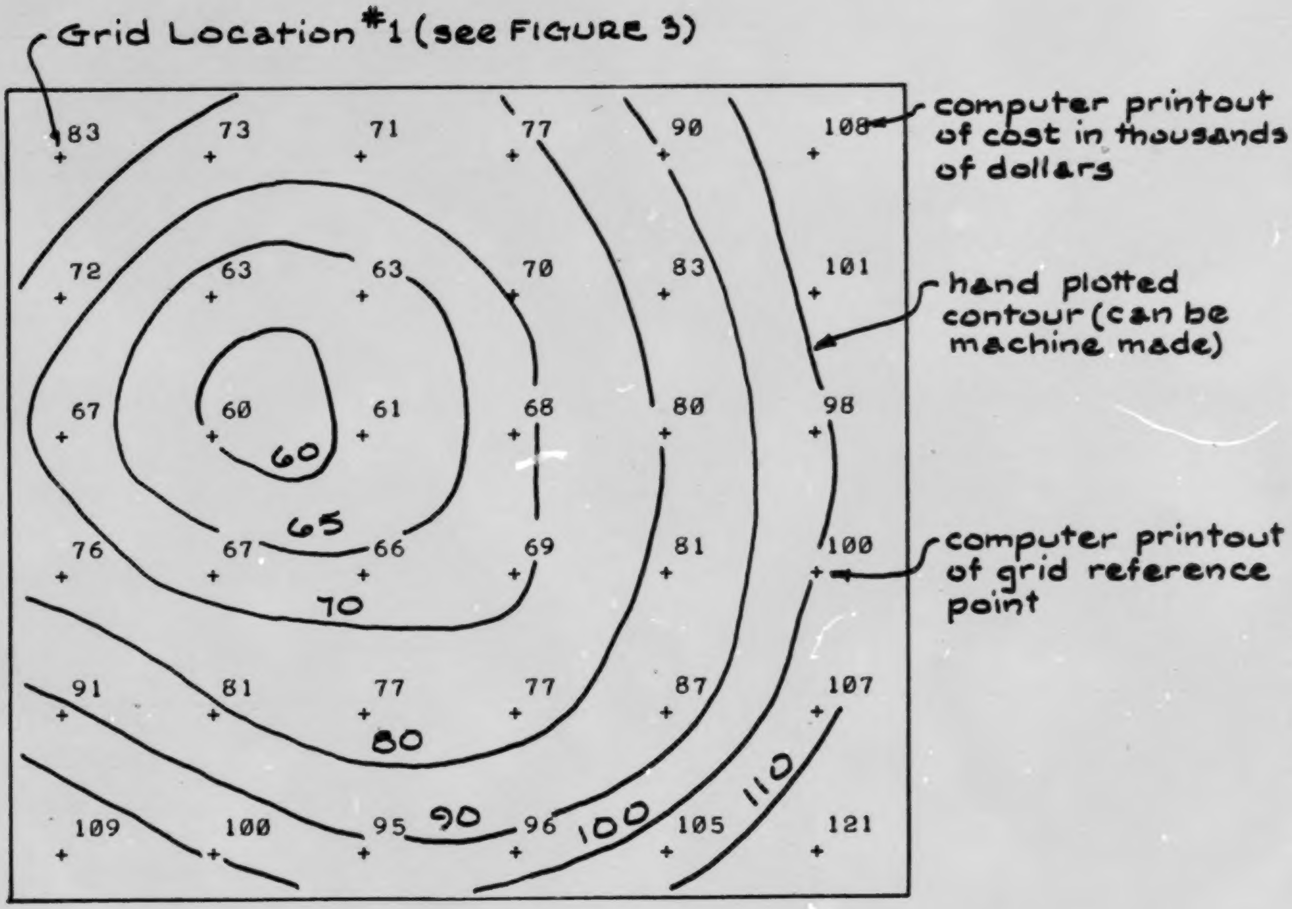
Conclusions and Future Development

The role of the computer in teaching or in actually carrying out this type of engineering problem is quite important, if not essential, in obtaining a complete solution. Perhaps the greatest educational benefit that has been observed is the attitude that the students seem to adopt after becoming "conversational" with the computer through their own programs. The concept of creating a parametric model by which they can establish a hierarchy among the factors, real or imagined, has been most enlightening. As a result, the students have become much more innovative in their approach to computer utilization and engineering itself.

Continued development of the data display formats will undoubtedly receive attention in the undergraduate oriented computer services and plans are underway to supplement the existing facilities with further means of plotting graphical output.

REFERENCES

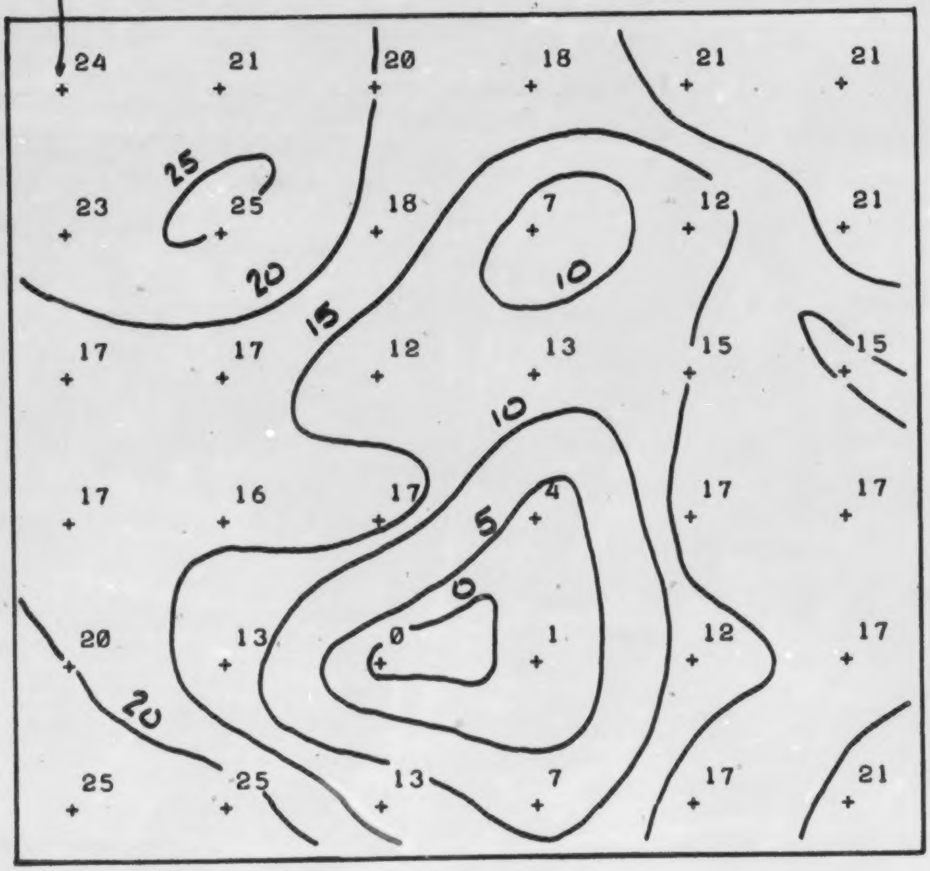
1. Rubenstein, Harvey M., A Guide to Site and Environmental Planning, John Wiley & Sons, Inc., New York, 1969.
2. Lynch, Kevin, Site Planning, MIT Press, 1962.
3. Schilling, Charles H., "Optimization in Site Selection", Computers in Engineering Design Education, Vol. III Civil Engineering, Univ. of Michigan, College of Engr., April 1966 pp. 134-146
4. Efroymson, M., and Ray, T., "A Branch and Bound Algorithm for Plant Locations", Operations Research, Vol. 14, 1966, p. 361.
5. Kuhn, H., and Kuenne, R., "An Efficient Algorithm for the Numerical Solutions of the Generalized Weber Problem in Spatial Economics", Journal of Regional Science, Vol. 4, No. 2, 1962, p. 21.
6. Gustafson, H., Gopinath, K. P., and Kramer, G. C., "A Weighted Aggregate Approach to R&D Project Selection", unpublished report, University of Wisconsin, 1969.
7. Davidson, Fredrick, "Utility and Decision Theory in a Planned Environment," unpublished report, McIntire School of Commerce of the University of Virginia, 1970.



CONSTRUCTION COST CONTOURS

FIGURE 4a

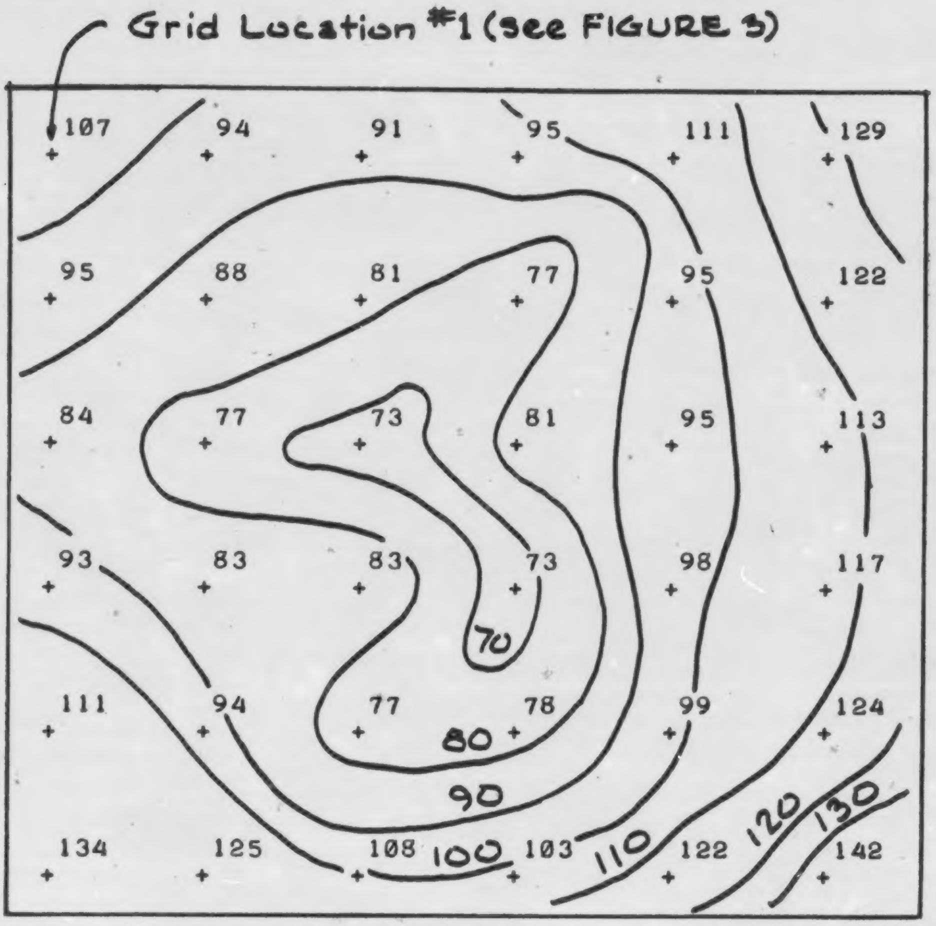
Grid Location #1 (see FIGURE 3)


24
21
20
18
21
21
25
23
25
18
7
12
21
20
15
10
17
17
12
13
15
15
10
17
16
17
4
17
17
5
0
20
13
0
1
12
17
20
25
25
13
7
17
21

same notes as on FIGURE 4a

SOCIAL COST CONTOURS

FIGURE 4b


107
94
91
95
111
129
95
88
81
77
95
122
84
77
73
81
95
113
93
83
83
73
98
117
111
94
77
78
99
124
134
125
108
103
122
142
70
80
90
100
110
120
130

Grid Location #1 (see FIGURE 3)

same notes as on FIGURE 4a

OVERALL SITE LOCATION

FIGURE 4c

UNDERGRADUATE EXERCISE USING A COMPUTER CONTROLLED LABORATORY MODEL TO OPTIMIZE A TEMPORARY CARGO STORAGE PROBLEM

Charles A. Moore
University of Illinois at Chicago Circle
Chicago, Illinois

ABSTRACT

A laboratory-computer interactive instructional apparatus is applied to the solution of the problem of the temporary storage of cargo on a wharf structure. The objective was to optimize the storage sequence so that the structural safety of the wharf is always maximized. The project described is one in a sequence of educational exercises to allow students to integrate optimal design with actual performance of real engineering structures.

The wharf structure was assembled by the student in a versatile soil-structure modelling tank. The cargo loading was applied under student programmed computer control. The loading sequence was computed based on the actual performance of the structural system during the loading phase. The criteria for controlling the loading sequence were preprogrammed by the student.

The principal educational objective of these exercises was to allow the undergraduate to comprehensively treat a real soil-structural engineering problem from conception of the original design through specifying the usage mode and observing the actual performance of the completed structure. Such experiences allow the student to bridge the gap between the design process and the behavior of the prototype.

INTRODUCTION

A problem commonly faced by teachers in fields such as soil-structural engineering is that field structures are far too large to allow the student to comprehensively treat the interrelationships between the design process and the actual performance of the completed project. Design procedures are often recommended on the basis of successful prior applications or because they comply with a particular design code. The student rarely has the opportunity, while in school, to evaluate the applicability of a design procedure based on the actual performance of the finished product. This restriction is often detrimentally carried into the philosophy of the design process itself. Thus the design is often based on a set of given, and often rather arbitrary, inputs such as building loadings. While this procedure results in neat, well defined problems for teaching techniques of analysis it all too often causes the student to isolate the design process itself from the overall context of the performance of the engineering structure.

These obstacles have been partially overcome through the use of a laboratory-computer interactive instructional apparatus designed for use in the soil engineering curriculum in the Department of Materials Engineering at the University of Illinois at Chicago Circle and previously described by the author (Moore,1971). The physical equipment consists of a container of soil in which a variety of scale models of engineering earthworks can be constructed. This construction is done by the student with whatever assistance he may require from the departmental machine shop. In addition, the capability to impose loadings on the system and to measure system performance under closed loop computer control is fundamental to the approach. The loading modes, the performance parameters to be measured, and the transducers to be used to measure these parameters are all selected by the student. The entire process is controlled by an IBM 1800 process control computer. The transducer reading sequence, the loading sequence and the interrelationship between the two are controlled by programs written by the student. The entire physical apparatus is of modular construction to allow the student to assemble a variety of problems with minimum help from departmental technicians. Moreover, computer subroutines are available to allow the student to write control programs in a somewhat problem oriented version of FORTRAN II.

This paper describes a problem that was solved in the Fall Quarter of 1970-1971 by a senior under a four quarter hour undergraduate research course.

DESCRIPTION OF THE WHARF PROBLEM

The problem chosen by the student for study was that of the safe temporary storage of cargo on a wharf. The premise was that a ship, on coming into port, would unload cargo directly onto the wharf for temporary storage. The problem was to decide the optimum positions for placement of successive units of cargo in order to maximize the factor of safety of the wharf against failure at all times during the unloading process. The modeling for the problem is shown in Figure 1.

For this problem the stuctural safety of the wharf is considered to be dependent only upon the stresses in the sheet piling and in the tieback rod. The stresses considered in the sheet piling are bending stress in the extreme fibers and shear stresses along the sheet piling. These stresses were measured using electrical resistance strain gages cemented to the sheet piling as shown in Figure 2. The safety of the tieback rod is based upon the axial tensile stress in the rod. The tensile stress is computed based on the total rod load as measured by a special student designed load cell. It may be seen from Figure 2 that the tieback rod was actually modelled as a knife edge contracting compression member in order to facilitate construction of the model. Conventional analytical techniques were employed to determine the stresses in the structural members.

The placement of cargo on the wharf was simulated in the model by five loading strips placed behind the wharf as shown in Figure 1. Loads were applied to these strips by low friction pneumatic loading pistons. The air pressure to these pistons was computer controlled through a stepping motor operated air pressure regulator. All of the above apparatus is fully described in the paper previously cited. Figure 3 shows a flow chart of the physical arrangement of the apparatus.

EXECUTION OF WHARF PROBLEM

The execution procedure for the wharf problem as programmed by the student is as follows. Initially none of the l/ading strips receives any loading. To initiate the loading a small probe load is placed at loading position one. The induced stresses in the sheet piling and tieback rod are then read and the decrement in factor of safety against failure is computed. This process is repeated using probe loads at each of the other four loading positions. The decrements in factor of safety against failure for each of the five probe loads are compared and the loading position for which the decrement in factor of safety is minimum is determined. A small, permanent load representing cargo placement is then applied to the loading position just determined to give the minimum decrement in factor of safety. The entire probing and loading process is then repeated and the second permanent load is placed. This process is continued for as many loading cycles as desired and the final result of the exercise is the safest loading sequence for the temporary storage of cargo on the wharf.

The computer program written for the problem consisted of a main program of 103 statements and two special subroutines of 13 and 14 statements. Output was through an on-line IBM printer and consisted of load number, footing on which load is placed, and number of loads on that footing. The following information is also printed for each strain gage on the sheet piling: section number, change in moment, moment, shear, factor of safety with respect to moment, and factor of safety with respect to shear.

For this project, several repeat runs were performed and the final optimum loading sequence was determined by treating the results of the runs statistically. This optimum sequence is shown in Figure 4.

DISCUSSION OF EDUCATIONAL BENEFITS OF PROBLEM

The wharf problem was specifically designed to expose the student to a variety of different types of problems. Moreover, a significant portion of the educational program was to develop the student's ability to organize and optimize the execution cycle for the problem. Figure 5 shows a flow chart of the educational course followed by the student. It may be seen that the student was confronted with several types of obstacles. Problems in design and creativity arose in the formulation of the project, the design of the structural layout, the design of the instrumentation and the writing of the control program. Logistics problems were faced in assessing available apparatus and physical facilities, in the construction of unavailable apparatus, and in arranging for computer time for executing the program. Obstacles of personal manual dexterity were faced in instrumenting the transducers, in assembling the apparatus, and in learning to operate the computer in order to execute the project. Intellectual obstacles were overcome in formulating the structural analysis techniques and in writing the control program. Finally, human relations problems were encountered whenever the student had to depend upon efforts of persons (machinists, technicians, computer programming assistants) other than himself. In particular, the student learned to provide maximum lead time when he required the services of such people. Perhaps the most important educational benefit of all was the realization that the success of the project depended upon having the ability to overcome a variety of types of obstacles.

CONCLUSIONS

In a post-course discussion the student showed evidence of having benefitted greatly from his participation in this project. The major satisfaction seems to have come from the student completely formulating and executing a project on his own. The versatility of the apparatus makes it possible for each student to be able to create a unique problem rather than repeat exercises which have been performed by numerous students previously. Such an individualized approach is beneficial in that it promotes self reliance in the student in addition to developing proficiency in overcoming specific educational obstacles.

ACKNOWLEDGMENTS

The author would like to specifically acknowledge the work of Thomas S. Dranger, a recent graduate in the Stuctural Engineering Curriculum at the University of Illinois at Chicago Circle. Mr. Dranger was the first student to make use of the apparatus described and is responsible for obtaining the results presented in this paper.

REFERENCES

Moore, C.A. (1971), Use of Process Control Computers by Students for Laboratory Exercises Involving Optimization in Soil Engineering Problems, Soil Mechanics Publication No. 9, Soil Mechanics Laboratories, Department of Materials Engineering, University of Illinois at Chicago Circle.

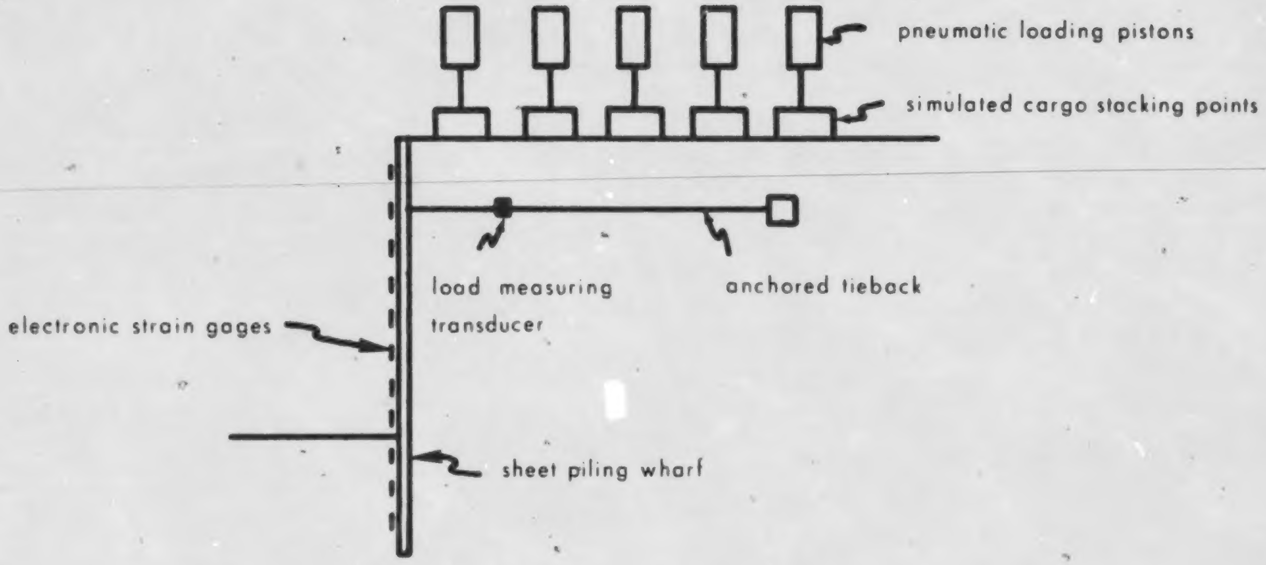
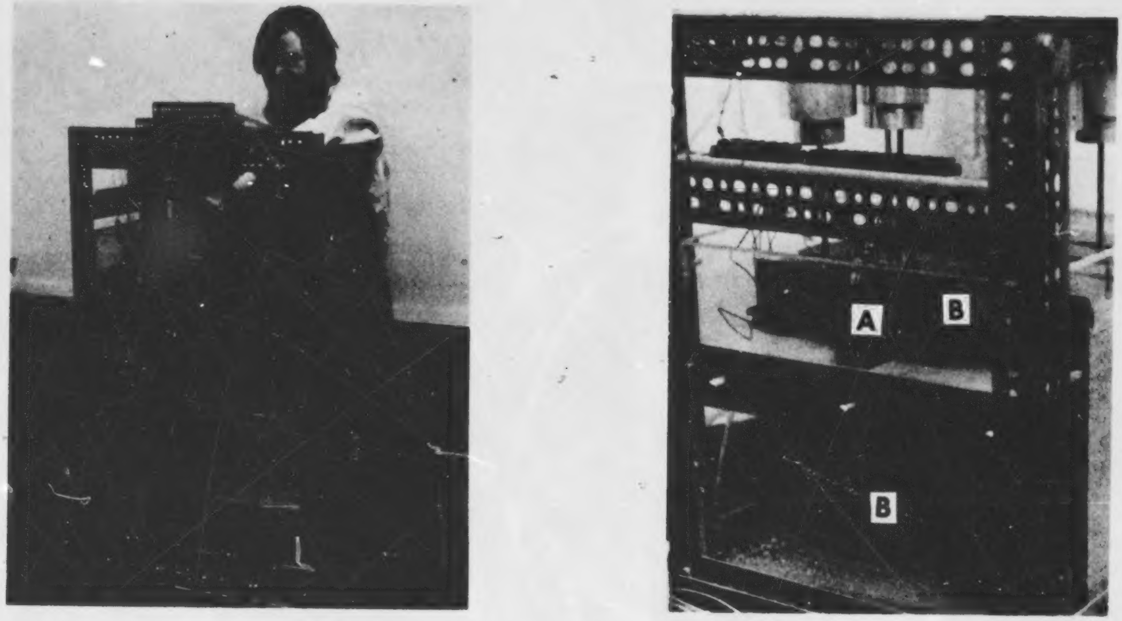

pneumatic loading pistons
simulated cargo stacking points
load measuring transducer
anchored tieback
electronic strain gages
sheet piling wharf

Fig. 1 - Schematic of wharf problem.



(a) overview of apparatus

(b) closeup of load cell (A) and strain gages (B)

Fig. 2 - Details of model wharf structure.

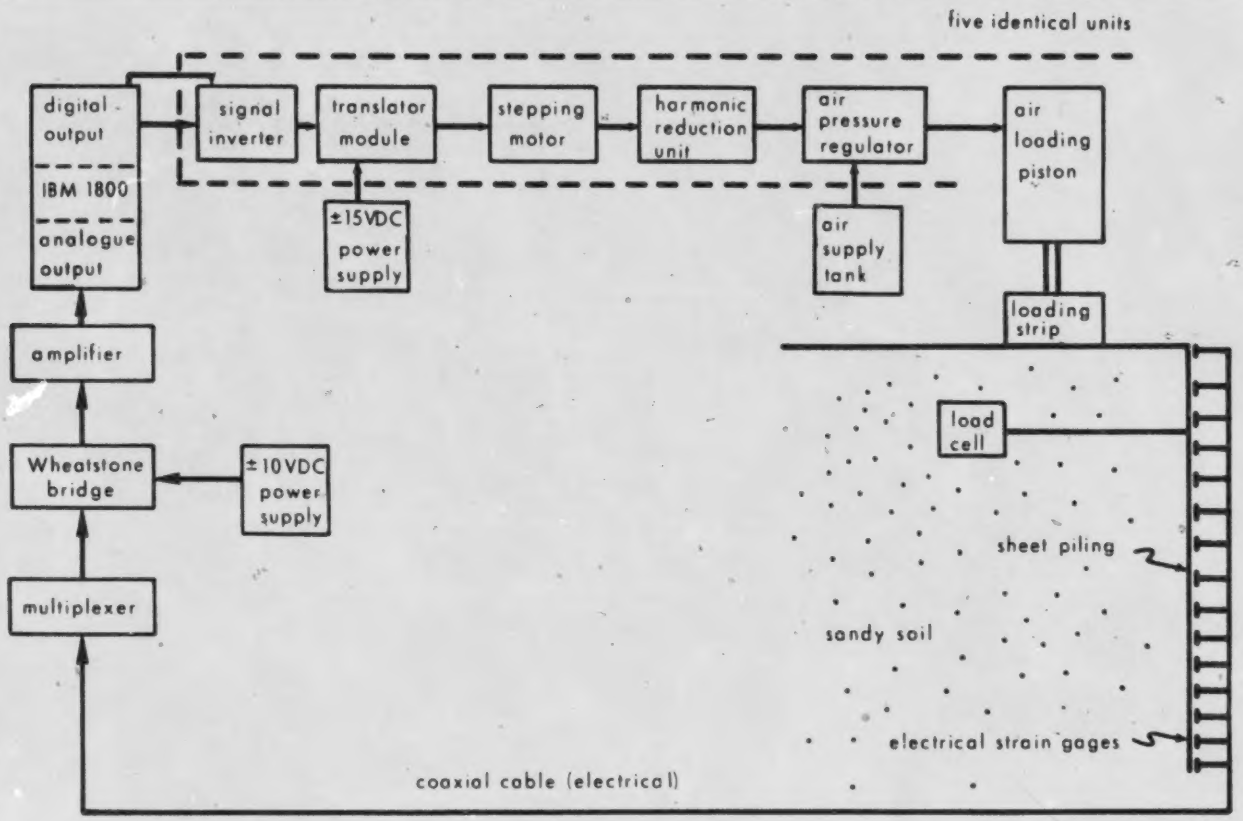

five identical units
digital output
IBM 1800
analogue output
signal inverter
translator module
stepping motor
harmonic reduction unit
air pressure regulator
air loading piston
±15VDC power supply
air supply tank
loading strip
amplifier
load cell
Wheatstone bridge
±10VDC power supply
sheet piling
multiplexer
sandy soil
electrical strain gages
coaxial cable (electrical)

Fig. 3 - Control flow chart for wharf problem.

411

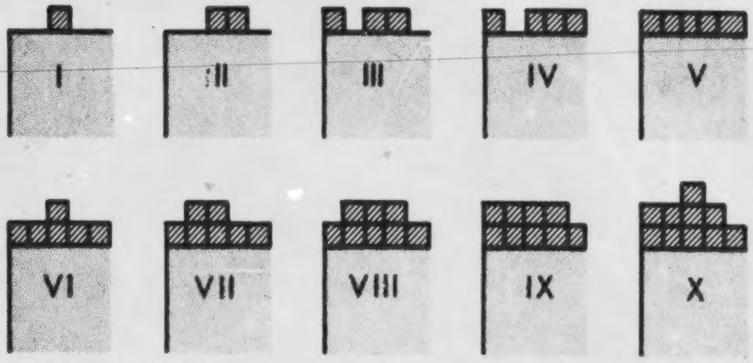

I
II
III
IV
V
VI
VII
VIII
IX
X

Figure 4. Optimum loading sequence for wharf problem

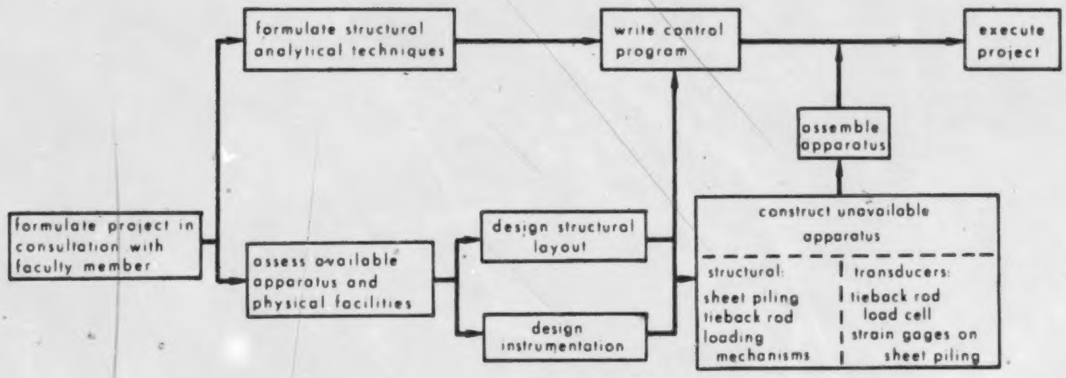

formulate project in consultation with faculty member
formulate structural analytical techniques
write control program
execute project
assemble apparatus
assess available apparatus and physical facilities
design structural layout
design instrumentation
construct unavailable apparatus
structural:
sheet piling
tieback rod
loading mechanisms
transducers:
tieback rod load cell
strain gages on sheet piling

Figure 5. Flow diagram of student learning process

USE OF A SMALL COMPUTER IN AN UNDERGRADUATE PSYCHOLOGICAL LABORATORY: A PROGRESS REPORT

David L. Sparks
University of Alabama in Birmingham
Birmingham, Alabama

The proposed usage of a small computer in an undergraduate psychological laboratory was described at the Conference on Computers in the Undergraduate Curricula hosted by the University of Iowa in June of 1970. This paper will describe our subsequent experiences with the actual use of the computer in a student laboratory.

The decision to purchase a small computer for our student lab rather than conventional psychological laboratory programming equipment reflects our belief that training in computer technology will become an essential feature of programs leading to a graduate degree in psychology. Additionally, computers are being used, increasingly, as psychological research instruments. The relative ease of interfacing the small computer with psychological research and the projected reduction in cost of the mini-computer point toward the eventual replacement of relay and digital logic programming equipment with the small laboratory computer. We hope, therefore, to introduce the student to many of the uses of computers in psychology early in his career while opportunities for acquiring skills in computer-related sciences are readily available.

In May 1970, our undergraduate laboratory was equipped with a PDP-8/I computer, manufactured by Digital Equipment Corporation (DEC). The basic package includes a 4,096 word, 12-bit memory and an ASR33 Teleprinter. To this we have added a TP-1351 Magnetic Tape Storage Unit (Tennecomp), four channels of analog-to-digital conversion, two channels of digital-to-analog conversion, a programmable clock, and a special interface which permits four inputs from each of five student stations and four outputs to each of these stations.

Thus far, the computer has been used primarily to control student experiments in our undergraduate course in experimental psychology. Each student station receives a standard electrical connector permitting four inputs to the computer and four outputs from the computer. The interface allows the computer to record lever presses, switch closures, interruptions of light beams, to deliver shock, solid or liquid reinforcers, and to control stimulus lamps, speakers or projectors.

Programs to control experimental parameters and to collect data from five student stations simultaneously have been developed. This is permitted by a program which allocates, on a round-robin basis, a one millisecond time slice to the program for each student station and an additional one millisecond time slice to the teletype which serves as an input/output device. At this point, four experiments have been implemented using the computer. With continued use, a library of programs will accumulate, permitting a wide range of experiments to be conducted depending upon the interests of the students and the instructor. Specific experiments utilizing the laboratory computer during the first year of operation are briefly described below.

Reaction Time Experiment. The purpose of this experiment is to compare reaction time under simple and disjunctive conditions. Under simple conditions, a subject is asked merely to respond to a signal as rapidly as he can. Under disjunctive conditions, the subject's response involves some choice on his part. An apparatus capable of presenting four stimuli and permitting three responses is used. The computer programs the stimulus presentations, presents a warning stimulus, randomly varies the time between the warning and response stimuli, records the latency for each response, controls the time between trials, and displays the latency for each trial upon request from the student

Sound Localization Experiment. This experiment investigates the effects of varying the time interval between inputs to the ears upon the perceived source of a sound. The computer is used to generate two rectangular pulses (.1 msec in duration), to control the order of presentation (left ear first, right ear first, or simultaneous) and to control the interval between clicks (0,200,400, or 700 microseconds). Seventy trials, ten of each type, are presented in random order. The apparatus consists of a set of stereo earphones, a dial calibrated from 0-180 degrees, and a record button. As each trial is presented, the subject points the dial toward the perceived source of the click and pushes the record button. The computer digitizes and stores this voltage value. Upon completion of seventy trials, each response value is converted into degrees and output on the teletype providing the student with a permanent copy of his data.

Operant Conditioning Experiment. A program allowing five operant chambers to be serviced simultaneously by the computer has been completed. The program is loaded into the computer by the laboratory instructor. As students enter the laboratory, they stop by the teletype and type "G" for Go; the teletype prints PY 60: UAB/OPERANT CONDITIONING EXPERIMENT to identify the program and then asks "WHAT IS YOUR CUBICLE NUMBER?". The student responds by typing one of the letters A-E. (Illegal charcters are responded to with a question mark). The computer then asks the student which phase of the experiment he chooses to run by typing PHASE?. The student responds by typing one of the numbers 1-4. One of the phases of the training procedure is a fixed ration program and the computer asks, when appropriate, WHICH RATIO?. Answers from 0-99 are permitted. When conversation is completed, the computer types, for example: PROCEED I HAVE PROGRAMMED PHASE 4 FOR CHAMBER E.

This particular experiment is based on Michael's laboratory manual for experimental psychology and permits the student to train his animal on a dark-light discrimination task. Phase I is a shaping procedure; Phase II, a fixed ratio; Phase III, extinction; and Phase IV, a discrimination problem.

Number of responses per minute for each chamber is automatically recorded and at the end of a session the student may request a print-out of this data by typing "T" for Type. Phases may be changed in one chamber, or data output from one chamber is permitted without interruption of data collection or control of the other chambers.

Pursuit Tracking Task. In pursuit tracking, the subject is asked to keep a response marker in line with a moving target. In this experiment, the target is a signal on a cathode-ray tube (CRT) display produced by a sine wave generator. The subject's response produces variation of a second voltage by means of a potentiometer, the output of which is represented independently as a cursor signal on the display. In this experiment, the computer controls the position of the target and cursor signals on the display using the analog-to-digital conversion features of the interface. As the task is presented, the computer is programmed to calculate an average error score for successive 30 second intervals, store the data, and type these scores at the termination of the experiment. The experiment is conducted with three different tracking signals which vary in frequency. The student is able to plot the learning curves of the subject, to determine the effects of variations in the frequency of the tracking signal upon performance, and to observe the development of predictive tracking.

In our view, two major advantages are derived from the use of the small computer in an undergraduate student laboratory. First a wide variety of experiments can be programmed with the small computer. In addition to serving as the programming equipment for controlling a particular experiment, the computer also serves as a data collection and analysis instrument. The capacity of the computer to collect, store, collate, and display large volumes of experimental data allows the instructor to assign experiments which would have been prohibitive without the computer. The student gains by being exposed to psychological experiments of current interest rather than old, but easily implemented, experiments. Secondly, the ability to program 10 or more student stations simultaneously will permit us to handle larger numbers of students in the laboratory. We can, accordingly, offer laboratories in courses which have been, primarily, lecture courses. We feel that these laboratory experiences are an essential feature of a strong undergraduate program in psychology.

The major disadvantage encountered thus far in using a small computer to implement an undergraduate laboratory is that the initial interface and programming problems are likely to be inherited by one individual. If conventional programming equipment were used, these responsibilities could be shared by other members of the department. We have tried to overcome this problem offering a non-credit course in programming to faculty and students. This has not been particularly successful since most of the faculty apparently lacks the time and/or interest necessary to acquire a facility for programming the small computer. It appears that we must either provide a programmer for the faculty or develop a stimulus-response programming language which is easily learned and which can be used to program psychological experiments.

Other proposed uses of the computer in the undergraduate curriculum will be described. Schematics of the laboratory interface and copies of the programs developed will be available for interested individuals.

DATACALL: A COMPUTER-BASED SIMULATION GAME FOR TEACHING STRATEGY IN SCIENTIFIC RESEARCH

Richard Johnson
Earlham College
Richmond, Indiana

One of the most costly forms of college level instruction today is that which occurs in science laboratories. Not only are there the tremendous capital investments in specialized equipment and space, but also the higher faculty-student ratio and the large blocks of faculty and student time demanded makes this a very costly form of instruction. For these reasons it is crucially important that we consider other possible modes of instruction which might meet some of the goals we have in teaching by laboratory experience.

One of the goals we have consistently stressed in our beginning courses in psychology at Earlham is to acquaint students with the way psychologists ask questions and go about attempting to answer them. We have felt that a part of understanding any field of man's knowledge involves not only acquiring a catalogue of the facts presently known in the field, but also beginning to grasp how man has come to know these things. For this reason, in the past we have involved our introductory classes in laboratory experiments . We were less interested in the students learning how to handle rats or manipulate the equipment in a psychologist's laboratory than we were in trying to get them to think like psychologists in attempting to answer questions. However, the usual introductory laboratory approach to this we found to be highly ineffective. Giving students "classical" experiments to "replicate" leads to student behavior patterns that mimic some of those associated with psychologists in the laboratory, but in terms of the goal mentioned above the result is not as clear. We discovered that some students resent collecting data to answer questions that they didn't feel were worth asking with information they could far more easily dig out of their textbook. Other students discover shortcuts to a laboratory report and it appears that most succeed in getting through the ritual of the laboratory manual without feeling the spirit of the experiment.

The problem with the replication approach to laboratory teaching is that students are rarely challenged to develop their own solution to some problem they have come upon. We next tried giving students open-ended problems and then challenging them to design a solution to answer the question. However, in terms of faculty time and the demands on equipment, this approach is even more expensive.

A partial solution to this problem began to emerge in the summer of 1968 when I designed a computer-based simulation game. The general idea behind the game is to give the student a simulation of some problem area in psychology and allow him to make some of the kinds of decisions that a psychologist would make in collecting data relevant to that problem. One of the main advantages offered by simulating a system is that some aspects may be left out. My simulation of psychological experimentation was built to leave out data collection. This means that the expensive equipment and special laboratory space commonly associated with data acquisition can be eliminated for this aspect of the teaching. However, there is an even more important advantage to leaving out data collection. The major expenditure of student time in the usual laboratory is involved with data collection rather than learning the logic of experimental design. While I do not deny the value of a student's "getting his hands dirty" by collecting "real" data, it is not at all clear that this experience should necessarily be a part of a student's initial introduction to experimentation. There are two issues which we might pause to consider before moving too easily to the conclusion that initial laboratory experience must involve data collection: 1) Efficiency in Instruction: Given a limited period of time how should the student's laboratory experience be allocated? In the typical laboratory situation the student will be expected to spend the greatest proportion of his time in data collection. If it is the case (as it is with most skills) that the learner needs to have a number of repetitions or trials in order to learn the strategy of experimental design, then the major period of time given to data collection may prevent adequate practice for the learning of experimental design. One way to meet this problem is to extend the period of time of practice over a number of terms or years as is commonly done with majors and graduate students. However, this does not meet the problem of the student who will not be able to take such an extended series of courses in psychology and this process may also be an inefficient way of doing this job even for the major. 2) The Programming or Design of the Learning Task: Is the best way to teach students how to do psychological research simply to start them doing it? Psychologists have learned a great deal regarding techniques for training lower animals to do very complex chains of behavior sequences. Some of the major findings that we commonly apply to lower animals suggest that when the task is complex enough, we should almost never give the animal the entire task at the beginning of the training and the temporal sequence involved in the task itself may not be the appropriate temporal sequence to be used in the training. If the skill we label "doing research" in psychology is a complex task, then it may be that we ought to break it down into manageable pieces and work out the appropriate training order rather than merely starting the student doing something and hoping that by a large number of repetitions he will learn. It may be pedagogically more sound to introduce students to the general logice of problem solving in psychology before they become too deeply involved in learning the intricacies of data collection for a specific experiment. I have had students in the past who focused so completely on how to handle rats, or use a particular piece of apparatus or the enormous problems involved in collecting a particular type of data, that they lost all sight of how they planned to develop conclusions from any data which they might collect. From this I am suggesting that students start by learning the related logic of the beginning and end points of any experiment, i.e., the way the planning which we call experimental design links to and determines the conclusions which can be drawn from any set of data. Once students have this firmly in hand, then we should be able to proceed to data collection.

Simulation offers a further advantage beyond the efficiency of introducing students to the decisions involved in experimental design. Because the simulated system with which the student is interacting has been built artificially, it can be closely controlled by the instructor. Situations can be developed which would not be feasible to explore in the normal laboratory situation and the simulated system can be adjusted to grade the difficulty of the problem and lead students to explore particular types of materials, or particular types of experimental designs. While all of these things are possible using natural systems and real experiments, it is much more difficult to develop the right experiment and typically only a few such experiments can be explored within a given period of time.

A variety of computer programs can be written which will simulate phenomena in psychology. The program described here sets up a system of six to ten variables in the computer which combine additively to produce an output. The sequence of events for a single round of a DATACALL game can be followed by looking at the schematic diagram at the end of the paper. After students are introduced to the idea of this computer game, they are given a description of a problem (called the scenario). An example of a possible scenario might suggest that the computer was going to simulate the teaching of mathematics to a high school population. The problem given the student might be to determine which factors are most important in leading to better learning as measured by a final exam in the subject matter. The scenario might then go on to tell about some research in this area and list a number of variables which are available in the program for the student to manipulate in his study. These variables could include: different techniques of teaching mathematics such as drill and practice vs. discovery techniques; different teacher characteristics; student characteristics such as sex, age, past knowledge of mathematics, etc. A student, after reading over the scenario description, decides upon an experimental design which consists of his choices on some subset of these variables. For each piece of data the student has requested, the computer first draws a starting value from some random distribution representing the individual variability. The effects on this "individual" of each of the variables in the experimental situation are next determined and added to this basic number. For any variable on which the student has made a specific decision, the computer puts in the appropriate effects, e.g., if one of the teaching techniques leads to greater rates of learning, then this will be reflected in the effect entered if the student chooses that technique. For any variable on which the student has made no choice, the computer draws from a probability distribution representing the possible values for that variable. For instance, if the student has made no choice on sex and the probability of drawing a female is .62 in the simulated school population, then the computer will draw randomly and enter the effect of whichever value it has drawn. Each time the computer goes through this sequence, it produces one piece of data representing one student's score on the final exam. The computer loops through this sequence the number of times the student has specified in the number of pieces of data he wishes to draw. The print-out that the student receives on this simulation describes his experiment to ensure that there were no errors in punching, and gives him a list of scores on the final examination in the hypothetical course. His next job is to analyze this data to see if the variables he tested had any effect on the outcome. Under these conditions, the student's task is to determine which variables have effects and how big they are. He does this by deciding which variables he wants to control and what values he will set for them. The set of decisions that he turns in constitutes his experimental design, and these decisions, as they interact with the computer program, determine the output.

Another way in which this approach to laboratory teaching differed from more traditional methods was that the computer simulation mentioned above was set up as a game. Games have been used for some time to teach military strategy, the effects of various kinds of international political decisions and the outcomes related to various business decisions. The advantages of the game format seem to be two fold: 1) since games have goals, the activity of the players can be focused along particular lines and as a corollary to the existence of goals, there is a built-in feedback system so that players know how well they are doing in relation to meeting the particular goal, i.e. "am I winning the game?". 2) Games have artificial rule structures which can be used to constrain the activity of the players along desirable lines and to explore the implications of a rule structure as an axiom system covering some field of human interaction (i.e. do the rules made explicit in the game mirror real constraints in the simulated system, and what are the implications of these constraints on how the system operates?

The DATACALL simulation was turned into a game by setting up a cost or ante which was associated with each decision made by a student and a corresponding payoff for the kinds of information that could be derived from the data of the experiment. Specifically, for any experimental design turned in there is a cost which increases directly with the number of pieces of data the student calls for and the kinds of limitations he puts on the variables in question. If he is able to discover a significant difference in the data he receives, he gets a payoff of 200 points. For a non-significant difference he receives only 40 points. Thus the general strategy he needs to adopt is to spend the fewest points in making his decisions consistent with obtaining the maximum information and payoff.

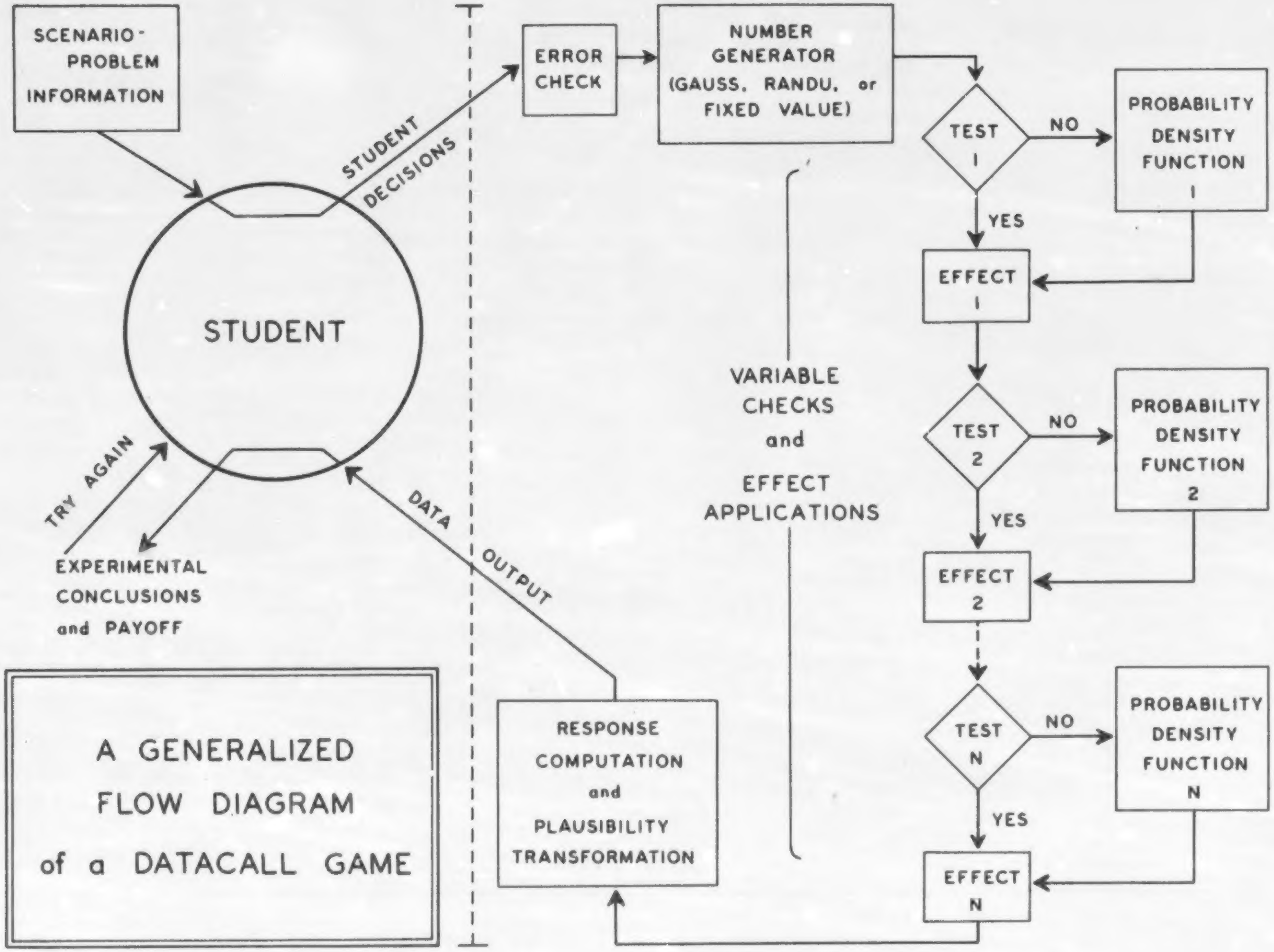
The way the game is set up, an individual student is "playing against the computer", although any number of students can play the game simultaneously in parallel. While this format of game play would appear to be ideal for interactive computer processing, we have had little experience in this mode thus far. Instead, the students have been free to turn in experimental designs at any time and then these were punched into cards by Lab assistants and batch processed so that the data was returned on the next class day. In this fashion a student has automatically assessed the ante for his experimental design when he turns in his set of decisions. However, as we played the game last year, no payoff was given the student until he handed in a single page laboratory report on his "experiment". We then posted these laboratory reports in a lounge area so that all students in the class had access to all other students' "published" results. This led to a great deal of student interaction in discussing results of various members of the class and in determining which types of designs are better and what questions ought to be asked next. To insure that students would not merely copy an already existing experiment "published" by another member of the class, a decreasing payoff function was set up for successive replications of the same experiment. This led to some very interesting student interaction. Some students wanted to wait for other students to do research so that they could read their "published" results before starting their own experimental program. However, this approach meant that ideas that they had on questions to be asked might already be researched and "published" before they got around to running the experiment. There was some complaining in the class at this time about my having established an atmosphere of "publish or perish". Another feature of having the students' reports posted was that one student, in reading the results of another student's experiment could try to replicate the experiment to see if the student's results were justified. We had set up an increased payoff for proving another student wrong in the claims that he made in his research report. We then had several students try to replicate experiments of others when they doubted the results.

There were several characteristics of the student behavior occurring in the play of the game that appear to be highly desirable from the point of view of learning to do research. 1) Students became very aware of the riskiness and tentative nature of experimental research and very sensitive to the fact that the kind of experimental design they set up has a great deal to do with whether or not they will be able to draw any conclusions. 2) Because any research report turned in had to have appropriate data analysis, all students became familiar with the statistical techniques of data analysis and got a great deal of practice analyzing data. Along this line I might mention two phenomena that we have never observed in introductory classes before. First, a number of students in the class went to the library and took out statistics books voluntarily in order to learn more about the job of data analysis. I had not suggested this and the only reason that I know that this occurred is that in at least two cases students came to me to ask me to explain a particular part of the book to them Thus, this approach to teaching seemed to function as a motivator for learning about particular techniques of analyzing data. Second, about one quarter of the class began using more complex experimental designs involving analysis of variance. After the game had gotten under way a number of students came to ask me to teach them analysis of variance so that they could design more complex experiments. Other students learned the technique by themselves. I also observed individual students figuring out specialized kinds of experimental strategies to try to reduce their costs. Some students ran pilot studies while others became very interested in reduced designs without my sending them to experimental design books to look at Latin Squares, etc.

The major advantages to this approach to teaching the decisions involved in research design are: 1) The rapid data acquisition of the simulated system allows many rounds of practice and more complex designs to be run than would be possible in an ordinary laboratory sequence. I have had students run as many as seventeen experiments in two weeks and other students who developed four way factoral designs. There is no question about the fact that students get a great deal of practice in both deciding on designs and drawing conclusions via statistical techniques. 2) The game seems to serve as a motivator for certain kinds of behaviors that might not have occurred otherwise. In attempting to cope with the demands of the game students find that statistical techniques suddenly become very relevant and they are motivated to learn more about statistics and how to use them. Others have also become very interested in the interaction of statistical techniques, kinds of designs and questions that they wish to ask. I have gotten into some lively discussions with some students concerning the whole model of hypothesis testing. All such outcomes I have never observed in introductory laboratories in the past.

While the games we have so far experimented with are relatively limited in scope, the general technique seems to be far broader. Thus, the approach may have applications in other science fields and there have been some initial experiments tried out at Earlham in both Physics and, more recently, Chemistry. It is also obvious that the simulation can be embedded in a wide variety of games leading to different kinds of student behaviors. It is possible, by the appropriate arrangements of payoffs and costs, to develop cooperative or competitive games and to get the students to focus on one aspect or another of experimental design.

It may be that the very best approach to use of this teaching tool will involve a type of game format with which we are presently experimenting. Because one of the most critical questions to be asked of any form of teaching by game is whether or not it will transfer to the real situation, we have recently become very interested in what I call forced transfer games. In this type of game the student is given a limited period of time in which to work with one game problem and then is immediately given a new game problem again for a limited period of time. The idea behind this approach is that the student, if he is to be successful across a wide variety of game problems, cannot learn merely the specific solutions to a particular game, but must also develop general game strategies. To the extent that the student can master this transfer problem and move to the final transfer stage, real experimental problems, this type of simulation gaming will have served its purpose to introduce in a more efficient fashion the ideas behind the strategy of experimentation.


SCENARIO-
PROBLEM
INFORMATION
STUDENT
STUDENT DECISIONS
ERROR CHECK
NUMBER GENERATOR (GAUSS, RANDU, or FIXED VALUE)
TEST 1
NO
PROBABILITY DENSITY FUNCTION 1
YES
EFFECT 1
TEST 2
NO
PROBABILITY DENSITY FUNCTION 2
YES
EFFECT 2
TEST N
NO
PROBABILITY DENSITY FUNCTION N
YES
EFFECT N
VARIABLE CHECKS and EFFECT APPLICATIONS
RESPONSE COMPUTATION and PLAUSIBILITY TRANSFORMATION
DATA OUTPUT
TRY AGAIN
EXPERIMENTAL CONCLUSIONS and PAYOFF

A GENERALIZED FLOW DIAGRAM of a DATACALL GAME

COMPUTER SIMULATIONS IN THE ELEMENTARY PSYCHOLOGY LABORATORY

Dana B. Main and Sabin Head
University of Michigan
Ann Arbor, Michigan

Background

This paper will describe the use of computer simulations as part of the undergraduate elementary psychology laboratory course at the University of Michigan. Psychology 210 is a two-hour course, consisting currently of 22 sections with a maximum of 14 students per section. The course is taught primarily by graduate student teaching fellows working with a faculty co-ordinator. Undergraduates typically take the course in their second year after completing at least one of two four-hour introductory psychology survey courses. The course is usually taken before statistics and is a requirement for an undergraduate major in psychology.

Elementary psychology laboratory courses generally have been modeled after laboratory courses in the physical sciences. The assumption has been that there exists a useful set of laboratory techniques that characterize psychological research. In a previous elementary course designed at the University of Michigan under such an assumption, the student was introduced to these laboratory procedures and techniques within the context of representative demonstration experiments. The student performed the experiments, practicing the prescribed laboratory techniques, and he wrote up the results in a laboratory report of a prescribed form. The types of experiments chosen for the course were typically those that could be performed within the time constraints of a two-hour class period. The student was rewarded for the care and accuracy in which he carried out his laboratory experiments.

At the University of Michigan the elementary course eventually departed from this more traditional model to encompass field and survey techniques. Indeed, the laboratory equipped with Skinner boxes and rat T-Mazes largely disappeared. Dr. Charles Morris and his teaching fellows developed a course consisting of four or five experiments in which students assumed more responsibility for the experimental design on each succeeding experiment (2). In many sections the final experiment was one completely designed and conducted by the individual student. The emphasis was on developing general research design skills which may or may not include specific laboratory techniques. Specific laboratory techniques with which the student gained experience were dictated by experimental design considerations rather than by a laboratory manual. The aim was to introduce the student to research in a wider spectrum of psychology. But, again because of constraints on time and facilities, the scope of subject matter by which the student could approach research problems and the types of possible research designs with which he could gain experience was limited. Further, it was virtually impossible to give a student experience in designing further experiments based upon his interpretation of the results of a previous experiment that he had conducted. In order to provide opportunities for sequential design decisions, and to expand the range of subject matter in psychology in which students could be given research problems to solve, time had to be compressed.

Shortly after Dr. Dana Main became faculty co-ordinator, a productive interchange of ideas began between her and her teaching fellow staff and with Dr. Richard Johnson from Earlham College. Dr. Johnson had developed a computer game, DATACALL (1) that he was using in his introductory psychology laboratory sections. The student was given a short description of a problem area, in this instance, verbal reinforcement. Manipulable variables were described and given code names and possible values. The student was required to generate a hypothesis or set of hypotheses from the problem area. Then, in order to test the hypotheses, he would select variables which he either held constant or systematically varied across experimental groups. Specifications for any desired experiment were submitted to the computer which then generated appropriate results.

Based on Johnson's work, one of the teaching fellows, Sabin Head, developed a set of similar programs in the summer of 1970, consisting of a series of modules that will be described below. These programs can encompass virtually any static multivariate model with little reprogramming. A program with a general capacity to accept new models directly or interpretively without any further programming is being planned.

It is hoped that a more general facility for building an expandable library of models can be developed in the near future. Graduate students are being encouraged to contribute models from their own spheres of interest and competence. This provides an interesting communication line between graduate and introductory students that is of an entirely different intellectual nature than the teaching fellow-student relationship. The role of the model-builder is being kept conceptually distinct from that of the programmer, in order to avoid the strong tendency to base decisions on programming considerations rather than on the models themselves.

Three teaching fellows, David Malin, D.W. Rajecki, and Susan Mueller have developed models that were used in the Fall, 1970 and Winter, 1971 terms. Malin's model deals with the genetic versus environmental bases of schizophrenia. The subject matter of Rajecki's model is imprinting. Mueller's model is on the motivation of factory workers in an industrial setting. These models reflect the research interests of those graduate students.

In the Fall, 1970 term, the simulations were administered through batch processing. That is, all experiments from students across all sections were submitted to the computer at one time each day and the output was available for the student the next day. In the Winter, 1971 term, each student was given an account and submitted his experiment to the computer and received his output directly.

Programming Considerations

Unlike many other uses of simulation, this use has the sole purpose of providing a microcosm for the student experimenter to explore. There is no strict requirement that the simulation be an accurate representation of any particular real world phenomenon, although the results should be reasonably plausible. Indeed, many controversial problem areas in psychology do not yet have "phenomena" in the sense of stable and consistent data, so any strict accuracy requirement might be out of place.

Program development was guided by the following objectives: 1) simplicity of operation from the point of view of the student totally unacquainted with computers; 2) similarity of operating details across both models and experimental designs; 3) flexibility with a goal of no restrictions on possible types of models or experimental designs; and 4) ease of generalization to new models.

All of these objectives have been met to some extent, but compromises on generalization, because it conflicted with operating efficiency, and on flexibility, to the extent that the conception was limited to static rather than dynamic models, were made. Both of these areas remain candidates for future development.

To a large extent, the objectives mentioned above were met in the conceptualization stage that preceded actual programming. In this stage we borrowed extensively from the ideas, programs, and experience of Dr. Richard Johnson at Earlham College. The following are the viewpoints arrived at in this stage.

Experiments

For this case study, the most fundamental form of an experiment is a set of separate runs or "conditions" each of which consists of observations on a so-called dependent variable. During a single run, some set of manipulable variables is specified and held constant. If they are changed from one run to the next they are often termed independent variables; if they are held constant they are often termed controls. Within a single run, however, this last distinction vanishes and all manipulable variables function alike. Individual runs are the building blocks for even the most complex experimental designs.

Models

For this case study the most general form of a model is a probabilistic function with several arguments. The arguments are simply the manipulable variables themselves. Usually models are impoverished in the sense that in the real world some unknown and/or uncontrollable factors may also play a role in determining the outcome of the experiments. Hence the dependent variable is not conceived of as an exact number but as a random variable, a distribution for each particular run. For example, if all variability is Gaussian or normal, the model provides a mean and standard deviation for each individual run. Observations are then simply samples from such a distribution. In a complex experimental design, therefore, each cell would consist of samples from a different distribution.

Variables

Most manipulable variables of interest can be simply categorized. Some variables take on only a finite and small number of states, each with a particular probability. Others might be continuous and exactly specifiable. Still others might be continuous but specifiable only within a range or interval. Thus the basic manipulable variables fall into three types: discrete, continuous, and range-limited. Each type of variable takes on a certain value in a particular run, and each value has a specified individual effect on the dependent variable. Interactions exist in the same way as these individual effects except that they are associated with the specific input values of more than one manipulable variable. The desired simplicity, standardization, flexibility and generalization has been gained at some cost to the user, though this cost is negligible in our context. Two such costs are direct results of the decision to model one run at a time. The first cost is that the simulation routines themselves can automatically provide only those statistics that can be calculated on a single run basis; other statistics are the user's responsibility. The second cost is that, while complex experimental designs are straightforward, they must be specified run by run, which may entail many runs. As an example, a three by three by three factorial design contains 27 cells and hence requires 27 run requests, which are usually submitted simultaneously.

Features of the Programs.

In keeping with a modular approach, the first step was to prevent the models from being machine-specific. This meant that the programs were to be written in a higher-level language, in this case FORTRAN IV. The second step was to organize the programs into a hierarchy of major distinct functions. To this end, all preparation, bookkeeping and error checking is in a single main program that calls on a separate subroutine for input from the student experimenter and on another separate routine which incorporates the particular model. All routines share a common storage of variables, effects and temporary results. A set of small utility subroutines performs such mundane tasks as randomization, Gaussian or normal sampling, and minor statistics.

This modular approach is carried into the internal organization of the programs themselves. All programming is done with small, stand-alone blocks or packets of instructions dealing with a particular operation on a particular type of variable. Each block has a single entry and single exit point for all purposes. Major sequencing within the program is done entirely outside of these packets, so that the programs consist of a sequencing framework which is filled in with blocks of operations. Deleting features not desired amounts simply to deleting the relevant blocks. adding features amounts simply to adding new blocks, and only the new ones need be debugged. The model prototype can be prepared quickly and modified until it coincides with the initial conception. Only then does adjustment of the program for operating efficiency take place. Storage is optimized, undesired features are eliminated, and small segments of the program that are heavily used are made more efficient. Such tuning for efficiency can be carried to any degree, but usually a small amount of adjustment in a few spots will serve to eliminate the most pronounced inefficiencies. This approach is analogous to the custom building of complex equipment from existing subcomponents. In both cases, extensive development that could not be justified for a single use is justified quite simply by multiple use.

Evaluation of Effectiveness of Simulations

Three types of evaluation are being conducted. The first is course-wide tests which will be described below. The second type is an evaluation of the final research projects that the students completed about a month after the simulations. The third type is student feedback. These latter two will not be discussed here.

Testing Program

In designing a test evaluation program for the computer simulations the following rationale was developed. First, the evaluation instruments and procedures should be designed to tap the specific problem-solving skills that the simulations are thought to develop, namely research design strategy decisions.

Secondly, the instruments were to be in a form that could be taken by all 210 students whether or not their class had used the simulations.

Third, a student's test performance is to be reported in terms of changes in types of design decisions rather than in terms of "right" answers. In order to measure change, a pre-post test design was used.

In both the pre-/and the post-tests a problem area was defined and discussed. In Part I the student made decisions about designing the first experiment he would conduct on that problem. The second part of the test asked the student to explain positive and negative results, and to make decisions about designing the next experiment on the same problem. In the third part of the test the student made decisions on which of three experiments he would do next given the results of three former but related experiments. The types of decision choices actually made on the pre- and post-tests are being analyzed. The attempt is to describe the research strategies chosen by the students without imposing any particular resarch bias. In short, value judgements are deferred. The goal is to report the test results in a form that can be used and understood by teaching and research staff with differing design biases.

Further development of this type of testing is being pursued for the future in the hope that it will be useful beyond the current narrow context of computer simulations.

Initial Impressions from Psychology 210 Instructors

The following are some initial impressions offered by the instructors using the simulations.

Class discussions took new and interesting turns that were impossible under other teaching approaches. For example, the relative utility of large multivariate designs versus a series of simpler designs become a lively issue. Students often insisted that they be taught some statistics in order for them to be able to better describe their results, and many even searched statistics books on their own. Students criticized others' reports, not because a critique was an "assignment" but because there was a desire and need to obtain the information. Students sought to synthesize results by proposing hypotheses and often divided into two or three "camps" to perform the related experiments. In fact, a small scientific community was created with discussions and problem-solving experiences that were not possible before.

Finally, our two-semester experience with computer simulations has focused our attention on the time dimension. Total research time is compressed in simulated experiments in that the major time-consuming steps in experimenting are omitted. The steps are 1) making the arrangements for data collection 2) collecting raw data, and 3) organizing and coding raw data. These steps are, of course, important and students get this experience in a later part of the course after the simulation work is terminated. However, when these steps are omitted students can generate an entire series of experiments within a six week period. The cumulated experiments for a class on a given simulation can be viewed as the literature or the history of the problem in the scientific community, to which the students have themselves contributed.

The students are thus placed in a situation in which they must constantly up-date their knowledge and revise their opinions as new information is gained. By way of contrast, most other current educational efforts direct the student to the past and some procedures require him to up-date his knowledge. By using computer simulations it is possible to deal with both in a very effective manner, while directing the student to look ahead.

Indeed, the exciting and conceptually different aspect of the educational process that simulations make possible is that they make future events meaningful to the student. This is one of the very few modes of instruction that has the potential of developing anticipatory or predictive skills. The student is placed in a situation in which planning and prediction have a real pay-off because the "future" is accelerated, if you will. The student can anticipate the outcome not only of a single experiment but of a sequence of experiments. His predictions of future events are quickly drawn into the present for evaluation and just as quickly become past history. Again, he has the opportunity to predict and the cycle begins again. This is, very simply, a feedback process. Skill in research design must encompass not only the sorting and processing of past information and the revision of present opinion, but also the development of anticipatory skills. Computer simulations challenge the student to look ahead and increase his opportunites for doing so.

The student's experience with accelerated time seems especially appropriate in view of today's world of ever-increasing change. Adaptability to a changing technology as well as a changing world can perhaps be enhanced by teaching modes, like the one described above, that challenge the student to anticipate and plan for the future.

REFERENCES

1. Johnson, Richard R., The Use of Simulation Games to Teach Decision Processes and Strategy in Scientific Research. Unpublished Report to the ESSO Education Foundation, New York, March, 1971.

2. Morris, Charles G., An Approach to the Introductory Laboratory Course. Annual Meeting A.P.A., Washington, D.C.; September, 1969.

INSTRUCTION IN STATISTICS: A REPORT ON THE COMPUTER LABORATORY FOR ANALYSIS OF DATA IN PSYCHOLOGY

Arthur Sandusky
Francis Campos
James Livingston
Elijah P. Lovejoy
David M. Messick
University of California Santa Barbara
Santa Barbara, California

For the past two years, the Department of Psychology at the University of California, Santa Barbara, has been using computerized instruction in our introductory statistics course, Analysis of Data in Psychology. In this paper, we will describe the problems we are attempting to solve with this system, the system itself with an emphasis on its more unusual features, and some of the conclusions we have reached regarding the effectiveness of our efforts.

A. Backround and Goals

Background

One of the more difficult curricular problems facing any Psychology Department is that of effectively teaching statistical and/or experimental methodology. Students decide to become Psychology majors because they are interested in the content of the field. It consequently comes as something of a shock to many students to find that they must take at least one course in statistical analysis before they will be permitted to partake of the more interesting substantive courses. Our students frequently failed to see the relevance of our statistics requirement. They frequently found the course dull and uninteresting. The importance of the ability to manipulate seemingly endless columns of numbers was lost on many students, and students who successfully completed the course often found it difficult to apply the techniques they had learned to problems they encountered in advanced experimental courses requiring statistical analysis of real data. Consequently, when the University of California announced in 1968 that it was providing funds to support innovative projects in university instruction, those of us most directly concerned with the teaching of statistics in the Psychology Department saw the chance to develop a real-time computing system which could be used in the Laboratory sections of our freshman-sophomore data analysis course. The system we wanted to create was one which could alleviate, if not eliminate, the problems described above.

Goals

Specifically, we wanted to design a series of programs or units to be used in weekly laboratory sessions which would help us achieve the following goals:

(1) Most students enter the course with a feeling best described as dread. We hoped to ameliorate that dread and, optimistically, to convert it into interest and curiosity.

(2) One of the most deadly aspects of a statistics course is the large amount of "number-crunching" required. We wanted to keep this inherently dull activity to a minimum, in an effort to emphasize the concepts and logical structure of statistical inference.

(3) We wanted to attempt to make the course "relevant" in the following ways: Rather than ignoring the ability of humans to make intuitive statistical judgments (sometimes quite accurately), we wanted to engage these abilities in order to show that statistical procedures represent formal, public rules which allow one to check his intuitions. Furthermore, we wanted to be able to generate large amounts of data in contexts that would closely resemble real laboratory situations. We wished to give the students an intuitive, as well as a formal appreciation of the relationship between distributional parameters and observed data.

(4) Finally, we wanted to maximize the generality of the tecnniques covered in the course to include advanced courses in which statistical analysis would be required. This was done by providing a Researcher's Manual describing the use of our system for the analysis of real laboratory data. Thus students were taught both statistical techniques and also how to use those techniques in our real-time system so that in advanced courses both the techniques and the system could be used to analyze real data from psychological experiments.

In summary, we wanted to teach as non-threatening a course as possible so as to make the experience interesting, relevant, and as useful to the student as possible in his subsequent work. The computing facilities we had to work with, and the system which has emerged, are described in the following section.

B. Computing System

Culler-Fried System

The University of California at Santa Barbara On-Line-System (UCSB OLS) is one of the Culler-Fried systems (Sammet, 1969, Smith, 1970). It is a time-shared, on-line, interactive computing system which is run on an IBM 360/75. Users access it from remote terminals, each of which consists of a 96-key keyboard and a 5- or 11-inch CRT storage display oscilloscope. The keyboard is divided into two segments of 48 keys, each of which provides a direct separation of OLS language elements into operators (upper keyboard keys) and operands (lower keyboard keys). UCSB OLS is currently composed of a single core system of programs together with two sub-systems, Card-Oriented Language (COL), and Mathematically-Oriented, Single Precision, Floating Point (MOLSF) system. The two sub-systems are mutually exclusive, although it is possible to move with relative ease from operations under one of them to operations under the other. The organization of OLS requires that the user specify which of the two sub-systems he desires at the time he initiates interaction. Specification of a sub-system defines the activity of some operators, and operations of these define the activity of those remaining. Since the statistics user system is supported by the MOLSF sub-system, we will describe only the sub-system and the basic system in any detail.

Computing Capabilities

The basic system of UCSB OLS provides the user with a number of different computing capabilities; the context for each of these is established with a single keypress. SYST specifies the context within which a user may "sign on", do manipulations on his library of data and programs, and carry out similar housekeeping activities. TYPE establishes a context within which, among other things, lower keyboard keypresses will be interpreted as alphameric characters to be displayed on the CRT. LI provides an index arithmetic context, within which lower keyboard keypresses are interpreted as identifying storage locations for integer operands. LIST is an important mode declaration operator; it establishes a context within which all keypresses are put into a single sequential list rather than being executed as usual. This list may then be stored and executed later upon "mention" of the name of its location. This deferred programming mode provides for the construction of user programs, which may be assembled and stored as system files, and called up for execution at some later occasion. The statistics system is one such file of user programs (actually a number of files); its characteristics enable students to avail themselves of the system's computing power without the necessity of being competent OLS programmers. In short, the basic on-line system, together with the MOLSF sub-system to be described below, provide a means whereby mathematically unsophisticated undergraduate students may effect computations some of which would be utterly beyond their technical competence, even though the results of those computations would be quite understandable to them.

The MOLSF sub-system defines the operators of the upper keyboard so as to provide a number of commonly used mathematical functions, e.g., sine, cosine, log, exp and others. Along with operators which are used to create and manipulate operands, the functional operators provide a means for user interaction at all points of a significant computation process. When used in conjunction with operators defined by the basic system, e.g., LOAD, STORE, and concatenated into strings via the LIST mode, MOLSF operators may be used to construct approximations to arbitrarily complex functions, e.g., numerical approximation to the integral of an integrable function. The user of MOLSF may, at his option, choose to use any one of three different kinds of mathematical objects as operands: 1) single real or complex numbers, 2) lists of real or complex numbers, or 3) arrays of real or complex numbers. The procedure for operating on operands in MOLSF is to load the operand into an accumulator or working area, apply the desired operator to accumulator, and store the result in some location. This procedure is the same for all types of operands, with MOLSF configuring the accumulator and storage locations appropriately. Procedures are also provided for transferring numbers from one "level" to another as well as for generating data which may be used for operands. For example, the button ID, in the context of lists of real numbers, causes a list of real numbers, equally spaced from the closed interval -1,1 , to be placed in the accumulator. The pair of button pushes ID? in the same context causes a set of real numbers, pseudo-randomly selected from the open interval (-1,1), to be placed into the accumulator.

When real lists and arrays have been declared to be operands, MOLSF allows the user to display these graphically as ordinates against the system-supplied abscissa (an X-Y plane, in the case of arrays). Since the CRT is a storage device, lists (functions) may be superimposed for comparison; the statistics system makes heavy use of this capability.

User Programs

The statistics system is composed of user programs of four types: 1) programs whose purpose is to simulate random sampling from specified populations; 2) programs which do the equivalent of looking up a value of a function in a table; 3) programs which perform the repetitive calculations required to do analysis of statistical data; and 4) question and answer programs which provide students with feedback on what they have learned. All four types of programs are more or less interactive, thus requiring that the student involve himself in the computation they perform. Members of the first and second classes of programs capitalize upon MOLSF characteristics in rather interesting ways. It will be recalled that MOLSF provides lists of random numbers from the interval (-1,1); these can easily be transformed into a sample (list) from the interval (0,1). Then, by virtue of a formal probability-theoretical transformation called the "probability integral transformation", this sample from the uniform density on the unit interval may be transformed into a sample from any population whose density function is continuous (there are other ways to sample from discrete populations). The procedure is as follows: first form a numerical approximation to the desired cumulative distribution function by forming the set of pairs consisting first of values from the population and, second, the associated value of the cumulative distribution function. Next, exchange the first and second components of these pairs, thus inverting the distribution function. Finally, take a random sample from the uniform distribution (0,1) and linearly interpolate the sample values into the numerically approximated inverse distribution function. MOLSF provides a single operator, EVAL, which performs a linear interpolation to find the proper functional values of a list, used as an operand, relative to a set of pairs defining the function. The operator is exceedingly powerful, and it alone suffices to do the equivalent of a table lookup; one need only compute the distribution function for a particular random variable, form the set of pairs as described above, but do the evaluation of the operand without exchanging the components of the defining list of pairs. By means of this technique one may get the probability value for the statistical rejection region of a standard normal, student's *t*, or any random variable whose distribution function may be numerically approximated.

In general terms it may be said of the statistics system that it is highly flexible in its application, since all the user programs in it are modular, and may be used in whatever combinations suit the interests and inclinations of the teacher of the course. The system is also unusual from the viewpoint of its extendability; the dynamic programming capability provided by the LIST operator allows one to reprogram on-the-spot even in the middle of a class, if desired. That is, the statistics system is essentially open-ended; it may be embellished and extended according to the abilities and interests of the teacher, or of the student.

C. Instructional System

Program Functions

There are four basic functions of the programs which make up the instructional system for this course. The first kind serves as a straightforward computational aid. It does the arithmetic operation that would normally be done by the student, only more quickly and with less chance of error. One such program takes a list of data and computes the mean, standard deviation, range, maximum and minimum values, and finally orders the data so the student can find the median.

A second kind of program takes the place of the table usually found in statistics books. The computer requests, say, the t-value and the number of degrees of freedom and responds with a graph of the *t*-distribution indicating the location of the chosen value and the probability associated with that *t*.

Third, multiple choice questions are placed at strategic points to check on understanding. The incorrect answers have been chosen to be diagnostic of frequent errors. When the student responds incorrectly, he is directed to remedial exercises designed to correct his particular misunderstanding.

Finally, there are programs which *simulate* samples of data from various populations through procedures described earlier. These programs serve both as sources of data for concrete examples of statistical exercises as well as exposing the students to the reality of variable sources of information.

Many principles of statistics lend themselves neither to formal proof nor classroom demonstration. This system has the ability to take large numbers of samples from any specifiable distribution so that each student can witness these principles in action and be convinced of their plausibility.

Decision Making Programs

The first exercise to be discussed provides the students with experience in making formal decisions about parameter values based on data from simulated experiments.

In the example, it is proposed that the mean IQ of college students on the West Coast is 110. It is suggested that samples be taken from three colleges to see whether they lie above or below the average. Prior to the actual sampling, the student finds the critical values which will cause him to reject the null hypothesis that the mean IQ at that college is 110. Then the student uses the computer to collect a number of samples of different sizes from each of the three colleges. For each sample, the computer gives the mean IQ and the standard deviation of the sample and asks the student for his decisions whether or not to reject the null hypothesis. When the sampling for a particular school has concluded, the program gives the student the total number of times he rejected the null hypothesis.

When he has finished sampling from all 3 schools, the student uses the computer to find the true values for each school. He then analyzes his error probabilities as a function of the parameter value, sample size, and significance level. For one school, the mean IQ is exactly 110 so that the likelihood of Type 1 error can be seen to follow the significance level. For the other schools, the means are 120 and 130 so that the student can observe power curves for different significance levels and sample sizes.

In a correlations program exercise, the student is first familiarized with the relationship between the correlation coefficient and the scatterplot, a graph of one variable plotted against another. Then the student is tested and given immediate feedback to sharpen his perceptions of the relationship between this number and the data it summarizes.

In the first part of the lab, the computer allows the student to select a correlation coefficient between -1 and +1. The computer responds by graphing a typical scatterplot of 100 points for that particular correlation. The student practices with this program for a short time to familiarize himself with the behavior of these scatterplots.

The second part of the lab consists of *estimating* the correlation coefficients of scatterplots given to him by the computer. After the computer presents a scatterplot, the student responds with his estimate of the correlation value. The computer displays the true coefficient and proceeds to the next trial. The true coefficient is chosen randomly for each trial so that the student sees a wide variety of unpredictable values. When the preselected number of trials is reached, the computer displays a scatterplot of the student's estimates plotted against the true values.

Sampling Distribution Programs

We include a lab on the central limit theorem, which states that if the sample size is large enough, the sampling distribution of the means will be normal, with a mean equal to the population mean and a variance equal to the population variance divided by the sample size. Since the formal proofs for these claims are too advanced for an introductory course, the computer is used to show that these principles hold.

Three very different distributions are used to show the generality of this principle--normal, rectangular and skew distributions. For each of these distributions, the student uses the computer to draw 50 samples of various sizes. For each *set* of samples the computer displays a frequency distribution of sample means and the mean and variance of these sample means. The student inspects the frequency distribution to see that it approximates normality and compares the mean and variance of the sampling distribution to those of the population. By completion of the exercise, students are convinced of the plausibility of this most powerful theorem without recourse to formal proof.

D. Classroom Implementation

Study Groups

Study groups comprised of three students are assigned to the sixteen available teleputer consoles in the computer laboratory classroom. We have found that study groups are superior to individual study because of a variety of instructional interactions that take place between students. For example, when conceptual difficulties arise students can often pinpoint the difficulty other students are having and take care of the necessary proctor work themselves. To take advantage of the stimulating effect of competition, a number of the laboratory units have been devised for which the study group members are in competition with each other in decision making. For example, students generate informal decision rules for simulated data and make competitive inferences before the formal rules are explicated. This sort of interaction gets students to "worry" about the problems involved in inference rather than just accepting formal dogma.

We have found that the effects of these interactions are maximized with three students per study group. With only two students, one may dominate and control the keyboard; with four, the time on the keyboard per student is too small for maintaining the conceptual thread of a session. These conclusions are based on careful observation, but additional research in this direction is needed.

The groups are formed by self-selection and result in a fairly haphazard ability grouping. The effect of the large individual differences in a study group has not been well studied and a grouping formula might be useful. With similarity matching, slower students can be deprived of the aid of the mathematically more talented and learn little. If abilities are too far out of line, frustration may result. However, students are not assigned to permanent groups and the self-selection process may be the best way to form cooperative groups; it is certainly the easiest.

Laboratory Sessions.

The time spent in a lab session is about two hours. This appears to be necessary if students are to acquire and discuss a reasonably complete concept. Longer times are a strain on attentional capacities although the atmosphere is fairly informal and relaxed.

It is vital that students complete a basic amount of work during the session so as to form concept closure. Carry-over of assignments creates great difficulties. If a student attempts to continue a problem session in a later lab, it takes him twice as long to complete the assignment since he spends much time reviewing the operations and concepts encountered in the initial session. He also loses the value of any pre- or post-lab discussions. The work timing of a unit deserves critical attention.

Laboratory Handbook

Students prepare for each session by reading an assigned unit in the handbook for the computer laboratory. Each unit begins with a general statement of the concept to be studied followed by a series of problems designed to lead the student into formulating the desired insights.

The first unit is an introduction to the system operation and describes the function of user operation programs. A number of optional problems are included, designed to provide familiarization with the console and to develop proficiency in using the keyboard at an early stage in the course.

The remaining units of the manual have been structured to allow for flexibility in their selection and ordering by different instructions.

The units include:

Statistical Decision Making
Descriptive Statistics
The Normal Distribution
Sampling Distribution of the Mean
Test for the Mean with Small Sample
Estimating and Testing Proportions from Large Samples
Tests of Differences Between Two Independent Samples
Chi-Square Correlation

The problems associated with these units are of three types: 1) parameters exploration problems designed to lead to the inductive formation of concepts or to develop training in "eyeballing" data; 2) decision making problems with large amounts of simulated data, designed to empirically demonstrate formal decision rules; and 3) computational problems designed to develop practice in the appropriate application of various statistical methods. Each unit contains a set of mainline problems to guide the student in the development of the lab session topic. These are intended to be sufficient to assure that everyone attains a minimal level of mastery of the concept which is consistent with the time limits and can easily be completed in a single session.

In addition to these, two other types of problem sets are needed: 1) remedial problem sets for slower students; and 2) concept-exploration problem sets for the faster student.

Remedial problems can be made contingent on the errors on diagnostic questions posed at the end of mainline problems. These questions can be addressed to the students in the manual and multiple choice answers entered on the keyboard. The answers can be followed by directions to proceed to either remedial or exploratory sections.

The only difficulty we encountered in using the teleputer itself to present questions and instructions was in the slowness of message printing when the time sharing system was under a high demand state in terms of the number of simultaneous users. Much if not all of this sort of quizzing could be handled in the manual but the use of the teleputer to perform this function has great appeal. Questions can be formulated on the spot and changes or clarifying additions are greatly facilitated by having them computer-displayed. Also, the possibility of future scoring and tabulation of this type of quiz results by the computer is then possible.

Continuous testing is of great help to both the student and instructor. Students find out whether or not they got the point, and so do you.

Researcher's Manual

The final unit in the course is devoted to practice with the researcher's manual. This manual is a guide to user programs which can be used in the practical application of statistical methods to real data. With these programs, the student can load data lists, find descriptive measures, graph histograms and scatterplots and perform a variety of statistical tests. It also contains the statistical tables needed in interpreting inferential results. It represents the sum total of the statistical tools that a student is expected to acquire in the course of instruction.

As a final test of the success of the course, the student is given a description of a set of experimental paradigms with the resultant raw data and he is expected to be able to apply the correct descriptions and tests in the analysis and interpret the results using the Researcher's Manual with the teleputer system. Approximately 80% of our students meet this criterion without error.

Several of the teleputers are available to the graduates of our course at all times in the Psychology Building. They can in their future experimental work have access to high speed computer analysis of data, and, by our criteria, have the competence to perform a proper analysis of data in psychology.

E. Conclusions and Evaluations

Enrollment and Costs

Each quarter, we have between 90 and 100 students in our course. Our total bill at the computer center averages $4000 to $4500 per quarter. This figure includes computing time devoted to the construction of new units and the revision and updating of established units as well as the computing done by students in the classroom. Thus we are paying about $40 to $50 per student per quarter, or $2 to $2.50 per student hour in the laboratory. By any reasonable standard these costs must be considered very low.

Student Response

At the end of each quarter, students have been asked to evaluate the course generally and the computer laboratory specifically. While the response is predominantly positive, two types of criticism are sufficiently frequent to merit noting. The first source of dissatisfaction stems from the fact that the entire on-line system occasionally fails, resulting in considerable frustration among the students. Such system failures are beyond our control, naturally, but we have expressed our concern to the computer center and their efforts to stabilize the system have increased the system's reliability dramatically in the past year.

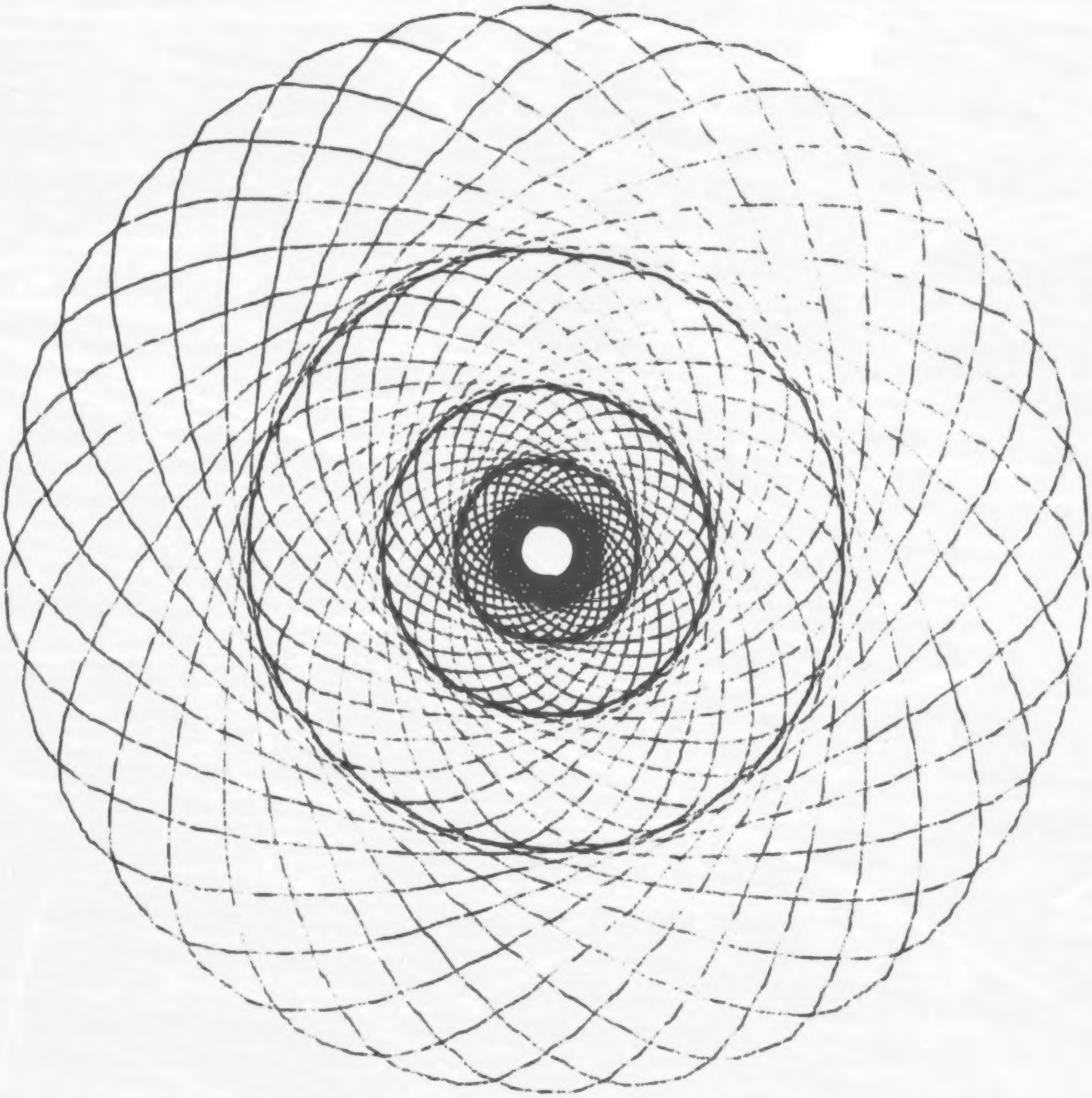
Another problem with which we are grappling is the tendency of some students to see the exercises as nothing more than tedious button-pushing. Some students find the laboratory so easy that it is boring to them. While this problem may be inevitable in large courses containing students with a broad range of talent, we are trying to minimize it by emphasizing the goal of the exercise as opposed to the details of the steps required to achieve the goal.

The preponderant response, however, is enthusiastic. Students comment on the interest and curiosity which the use of the system excites and on the enhanced insight and understanding which they achieve of the topics covered in the text and in lectures.

REFERENCES

Sammet, Jean E., Programming Languages: History and Fundamentals, Prentice-Hall, Inc.: Englewood Cliffs, N.J., (1969); pp. 253-255.

Smith, Lyle B., A Survey of Interactive Graphical Systems for Mathematics, Computing Surveys, Vol. 2, No. 4, December, 1970; pp. 261-301.



William Carney

USE OF COMPUTERS IN STATISTICAL INSTRUCTION

Richard N. Schmidt
State University of New York at Buffalo
Amherst, N. Y. 14226

1. INTRODUCTION

The purpose of this paper is to present some ideas concerning how computers can be used for enhancing concepts for students in a first course in probability and statistics.

The author has been teaching in the area of computers and statistics for many years, but the ideas in this paper represent primarily some of the outcomes of the experimental project on the use of computers in statistical instruction sponsored by the National Science Foundation and the University of North Carolina. During the 1969 workshop, the author was the leader of a panel that prepared two course outlines. The first course outline covered a first course in probability and statistics for students with only a high school mathematics background. The second course outline also covered a first course in probability and statistics, but was developed with the assumption that the students had had the equivalent of about one year of college calculus.

The author implemented this second course outline for advanced freshmen, all with a calculus background, in a Freshman Seminar titled "Statistics as a Mathematical Science in a Computer Environment". The following paper is drawn largely from some of the experiences surrounding this course. As far as it is known, although some thirty outlines were distributed, this was the only attempt to implement the course outline for such students.

2. COMPUTING FACILITIES

The computing facilities among universities are considerably varied ranging from hands-on small computers within departments, to complete networks of interactive terminals in a few universities. By and large however on the typical American campus students still are restricted to a batch mode of operation. In order to give the reader a perspective of the equipment utilized for this experimental course the following list covers the equipment available to the Department of Statistics at SUNY/Buffalo.

a. Statistical Laboratory. We have a small statistical laboratory in which we have 6 Friden 132 calculators, 2 Monroe Epic 2000 calculators, and 2 Olivetti Programma 101 calculators. We also have a Wang Model 700B, but it arrived too late for use by the students in this experimental course.

b. Terminal Room. Adjacent to the statistical laboratory we have a terminal room. In this room for the course we had a Model 029 IBM card punch, an ASR-33 teletype terminal, and an IBM 2741 administrative terminal system. The ASR-33 was leased through a contract we had with Graphic Controls Corporation. It was only through the support from NSF that we were able to have this terminal. Otherwise we should never have had a chance to use it. (Originally, we had planned the course around this terminal.) Via telephone, we could connect our ASR-33 to a GE 265. This computer had been acquired by Graphic Controls from Dartmouth and was the very one on which BASIC had been developed.

c. Computing Center. The University Computing Center is housed in a separate building a short distance from the Department of Statistics. The main computational machine is a CDC 6400, with 65,000 words of memory. (Actually for the demands on the system this memory size is far too small.) Also in the computing center is an IBM 360 Model 40 computer.

The 2741 mentioned above is connected with the 360/40. Its main function as a terminal system is to permit the creation of files and text editing operations. However through a University-built interface it is possible to use the 2741 for terminal program entry. In essence one creates a file--in this case a FORTRAN program and data--which then is transferred into the 6400 for computation and execution, and the output is then returned back to the terminal through the 360. The turnaround time using this sytem is from a few seconds to a half an hour depending on work load.

3. PROPOSED COURSE OUTLINE

Immediately following, in Section 4, is the course outline developed by the bio-statistics sub-group at the June 1969 NSF-UNC workshop. The code names for specific computer programs have been left as originally set down. Interested people can get complete information about them by writing to the Department of Biostatistics at UNC.

The outline below is introduced by two sections on overall objectives and broad topics. Then the detailed outline as proposed for each of the four broad topics is given.

After those of us who developed the outline listed below had a chance to review it after it was completed, we decided that it was too much for a single semester. In part 5 below I will cover some of my experiences in trying to implement the course as proposed by the outline given next.

4. THE COURSE OUTLINE DEVELOPED AT THE NSF-UNC WORKSHOP

In this section is the proposed course outline for a one-semester introductory course in probability and statistics for students with approximately one year of calculus. The outline assumes a computer environment.

General Objectives:

1. To involve students in computer programming and remote processing, with the objective of reinforcing their understanding of statistical concepts.
2. To prepare students for further coursework in their own fields.
3. To increase students' ability to critically read literature in their chosen field of study.
4. To prepare students for advanced work in probability and statistics.

Broad Topics:

1. Fundamentals of probability,
2. Distribution theory,
3. Theory of inference,
4. Analysis of experimental data

PART I: FUNDAMENTALS OF PROBABILITY

A. OBJECTIVES:

1. To prepare the student for the study of distribution theory and concept of a probabilistic model.
2. To enable the student to use specific methods (compute and manipulate) in probability.

B. TOPICS:

1. Experimental situations,
2. Abstractions of outcome space,
3. Events, independents of events, algebra of sets
4. random variables
5. Probability of an event. Marginal, joint, conditional, independent
6. Distribution of a random variable.

463

C. COMPUTER INVOLVEMENT:

1. Uniform random numbers,
2. Random walk,
3. Combinatorics,
4. Using a list of numbers to apply the frequency approach to estimating probability,
5. Demonstration of addition and product rules,
6. Use of straight line to show probability space and intersection,
7. Demonstration of runs and ranking,
8. Enumeration by loops,
9. Tree construction

D. EVALUATION OF STUDENT:

1. Conventional quiz,
2. Facility with remote ASR 33 terminal,
3. Simple computer programs,
4. Homework (where required).

PART II: DISTRIBUTION THEORY

A. OBJECTIVES:

1. To investigate empirical and theoretical distributions and their relations, one to the other.
2. To introduce analysis of distributions by employing transforms and other techniques.
3. To study the special distributions of importance to statistics.

B. TOPICS:

Discrete Distributions
(a) Bernoulli, (b) Uniform,
(c) Binomial, (d) Negative binomial,
(e) Geometric, (f) Hypergeometric,
(g) Poisson, (h) Other
Continuous Distributions
(a) Uniform, (b) Triangle, (c) Exponential,
(d) Normal, (e) Cauchy, (f) Student's t,
(g) Chi-square, (h) F, (i) Log-normal,
(j) Logistic, (k) Gamma, (l) Beta,
(m) Other
Areas to be Examined
1. Combine the treatment of the related theoretical, empirical, and sampling distributions.
2. Moments,
3. Joint distributions,
4. Marginal and conditional distributions,
5. Independence,
6. Use of statistical tables.

C. USE OF THE COMPUTER:

1. Plot densities and distributions (overlays)
2. Probability plotting
3. GENDAT, NORDIS and pseudorandom number generators
4. Central limit theorem
5. Tchebychev's inequality
6. Sum, and sum of squares, of normal deviates
7. Assign student a programming problem

D. STUDENT EVALUATION:

1. More emphasis on homework
2. Conventional exam on distribution theory
3. Mark on assigned programming problem

PART III: INFERENCE

A. OBJECTIVES

1. To teach students how to make decisions in the face of uncertainty.
2. To teach students how to construct models and to relate them to data.
3. To explain the difference between parameters and statistics.
4. To make (relate) probability statements about (with estimates) of parameters.

B. TOPICS:

1. General decision concepts, observation space, type I, II, and III errors.
2. Testing hypotheses.
3. Estimation (with a prior).
4. Estimation (without a prior).
5. Confidence regions.
6. Tolerance regions.
7. Nonparametric methods.
8. Assign a project due at the end of the semester. The project must involve the literature.

C. USE OF THE COMPUTER:

1. Calculation and simulation of confidence intervals.
2. Power curves.
3. Graphic displayes of confidence intervals.
4. Data smoothing and treatment of outliers.
5. Canned programs for estimation and testing.
6. Computer games (student vs. the computer).
7. Goodness of fit and adequacy of model.
8. Student to write a simple program derived from involvement in this subject area, such as:
    (a) Joint confidence region.
    (b) Likelihood ratio, chi-square, confidence region.
    (c) Maximum likelihood estimates.
    (d) Plot likelihood surfaces.
9. Any of several nonparametric tests.

D. EVALUATION OF STUDENT:

1. Interim report.
2. Problem-oriented test.
3. Student's performance on decision-making assignment.
4. Assigned homework (mostly related to the computer).

PART IV: FURTHER METHODS IN THE ANALYSIS OF DATA

A. OBJECTIVES:

1. To explain the meaning of the data, their treatment and display.
2. To have students use the computer to handle a set of data.
3. To describe data according to theories covered; to postulate various models and to test them using data, both real and computer-generated.
4. To cover the choice of methods specifically applicable to well defined experimental situations.

B. TOPICS:

1. Tabular presentations.
2. Graphical methods.
3. Descriptive statistics, Tchebychev's inequality, tolerance limits, data smoothing.
4. Curve fitting (Adequacy of model, goodness of fit, analysis of residuals, etc.).
5. Data transformations (See Draper and Hunter, Technometrics, May 1969).
6. Analysis of Variance.
7. Categorical data.

C. USE OF COMPUTER:

1. Packages for handling and analyzing data:

(a) EDIDAT
(b) UNIDES
(c) MULDES
(d) POLREG
(e) TRADAT
(f) PRODAT
(g) BLOCKS
(h) SQUARE
(i) ANATAB
(j) SCADIA
(k) ONEWAY
(l) TWOWAY
(m) ZEROES
(n) REGANA
(o) PARCOR
(p) MATCOR

D. EVALUATION OF STUDENT:

1. Term project.
2. Required computer and other assignments.

5. STATISTICS AS A MATHEMATICAL SCIENCE IN A COMPUTER ENVIRONMENT

In this section I should like to report on some of the actual experiences in trying to implement the outline given above, as a course given under the title of this section. As we who developed the outline anticipated, there was far more work involved than could be covered in one semester. In addition to this fact there were many problems with computers. I had intended to use the programs developed by the Department of Biostatistics at the University of North Carolina. Thanks to support from the National Science Foundation I was able to lease an ASR 33 terminal and negotiated a contract with Graphic Controls of Buffalo. As I mentioned above, the actual Dartmouth computer and library of programs had been acquired by this company. Unfortunately, even though I had been assured the FORTRAN programs developed at UNC would work perfectly on this system, there were certain problems with each program so that we had but limited use of the packages. Consequently, during the semester we used mainly a program to generate random variables (GENDAT) and a univariate descriptive program (UNIDES). However, by the time these were working properly we had gone on to other methods and thereafter made but limited use of the other package programs, although the original intention was to use all of them.

The course included both lectures and laboratories. The lectures were the equivalent to three 50-minute periods per week, together with collective and separate laboratory sessions. The seminar was restricted to twelve students, but even so, the sessions seemed to work best if only one or two students were involved with equipment at a time. Thus, the scheduling of equipment use became important.

As far as the approach was concerned, I immediately began with $(\Omega, a, P,)$ to introduce the concept of probability. Even though they were freshmen, they accepted the idea. Of course, I used only discrete examples. Essentially, the students appeared to be motivated mainly by the programming part of the course, which was conducted as follows: One group would be using the ASR 33 to write programs in FORTRAN. Then another group would prepare punch cards for the same program which we would then run on a batch mode on the CDC 6400. Concurrently, a third group of students would be entering the program through the 2741 remote terminal into a batch queue into the CDC 6400 via the 360/40. Output would come back to the 2741 in our terminal room. In this way students would get experience with different methods of programming and entering programs and would be able to compare the methods, the results, and the limitations. As far as the subject matter is concerned, we were able to complete part I (fundamentals of probability) and part II (distribution theory) as listed in Section 4 above. As one can see from the list of topics presented there, the subject covered primarily was probability. Unfortunately, time was running out, and although our objective was to concentrate on statistics, we only had about five weeks which we spent on inference, methods of data analysis, the calculation of moments, and some simulation using random numbers. Most of the course was covered by notes prepared by the author. As for texts, the students were encouraged to find whatever books appealed to them. During the semester, the students used over 15 different books for reference.

With regard to our computer programs, we eventually reached 55 pages mostly of programs in FORTRAN and computer outputs. Some of the topics covered in programs included: moving averages, quadratic formula, normal density, plotting of normal density, random number generation, computing and plotting normal distribution functions, random walk, examination of random digits for randomness, computation of descriptive statistics, plotting of random walks, calculation of moments, and illustrating the central limit theorem.

We also had some time for exploring the use of the 2741 for text-editing purposes, besides its use as an input-output device for remote batch operations.

6. SUMMARY OF THE EXPERIMENTAL SEMINAR

It is my feeling that a good deal more experimental work must be carried on in this area. I am not fully convinced of the merits of the remote terminal approach. Perhaps I am somewhat aware of the high cost of terminal operations and compare these with batch operations. I realize that the turnaround might be slow in some situations but I think there is more interchange among students in evaluating, reviewing, and discussing programs when they have batch process listings, than when several are gathered around a terminal.

If there are going to be large classes of students, then I think the terminal would be of limited usefulness if only one or two are available on campus.

With regard to the 2741 as a remote terminal, I approached its use with great enthusiasm but found that it also had severe limitations as compared to batch operations. Even for my personal use of the computer, although I have a terminal across from my office, I still find it more convenient to walk to the computing center with my decks of programs.

7. APL

One of the fascinating outcomes of the 1969 workshop was the enthusiasm of Dr. Mark Appelbaum and Dr. Frank Anscombe in introducing us to APL. Although we have a system at Buffalo that can handle APL, its time is so heavily scheduled that it was only recently that we were able to get on the system--and that time started at midnight. Recently we have acquired a batch APL facility for our CDC 6400, but so far this batch program seems still in the development stage.

With regard to APL and a first course such as the one described above, a special note should be made of the recently available preliminary manuscript, A SHORT COURSE ON COMPUTATIONAL PROBABILITY AND STATISTICS, by Ulf Grenander and Walter Freiberger of Brown University. It looks like a fascinating set of notes. The computational framework is APL-oriented and could set a trend for the future.

8. OMNITAB

Even before coming to the 1969 workshop I had been trying to get OMNITAB on our system. Following the enthusiastic recommendation of Dr. David Hogben, I tried for several months to get it implemented, but finally gave up because of the amount of work required to get it segmented for our CDC 6400. This language has great potential for data analysis, but it was not available for my experimental course.

9. OUTLOOK FOR THE FUTURE

Without question the computer is having a great influence on the teaching of statistics. My feeling at present is that we are still exploring economic ways of using the computer on a mass basis to optimize the effectiveness of our teaching at a low enough cost to make use of the computer feasible for introductory courses for large numbers of students. As an example, I know that my students in the experimental seminar did the kinds of things that would have been completely impossible without the computer. For example they first proved the Central Limit Theorem mathematically. They then simulated the drawing of large random samples from various kinds of distributions and showed how the empirical curves began to approach the normal (0,1) curve as n increases from 1 to 2 to 5 to 10. Such simulations are very powerful in exhibiting the real meaning of the theory. When the student has to program the simulation, he really must understand the symbols that often are hidden within a mathematical context.

In conclusion, I should like to say that the course was very enjoyable. However, one who attempts to initiate such a course should be well aware of the tremendous amount of development work that will be required. He should be prepared for considerable frustration in the computer area. He should plan for a high cost in time and money in searching for, developing, preparing, duplicating, and distributing materials for the students.

INCLUSION OF EXPLANATORY MATERIAL IN COMPUTER PROGRAMS FOR STATISTICAL ANALYSIS

Jolayne Service
North Carolina State University
Raleigh, North Carolina

A student of elementary statistics is often overwhelmed by unfamiliar concepts, terminology, and notation. He may learn to perform calculations (or learn to make a computer perform calculations), only to find that the resulting numbers, purported to be the "solution" to the problem posed, have little or no meaning to him. The discussions of statistical techniques in textbooks may be too general for him to apply effectively. He thus may leave the statistics course concluding that its X_i's and σ^2's are irrelevant to his own field and that statistical procedures are likely to confuse rather than illuminate his research.

To ease such a student's difficulty, we are including explanatory material in our set of "canned" computer programs for statistical analysis. This material is designed to discuss the analysis performed in terms of the data actually submitted to the program.

CONTEXT

The Department of Statistics at North Carolina State University has used a set of canned programs for the IBM 1130 computer for the past four years. Written primarily for use by students in undergraduate and graduate statistical methods courses, these programs are available to any faculty or staff member or graduate student and to undergraduate students taking statistics. Input and output of each program are described in standardized write-ups. Students enrolled in statistics courses that include use of the programs attend an orientation session at the beginning of the semester. The session includes discussion of programs available, instruction in using the write-ups, and, when necessary, instruction in keypunching. A student can then prepare an input deck and submit it to the computer operator, who runs the program and gives the student his output. The operator is familar with the programs and can usually help the user whose run failed.

Each program is written in FORTRAN and stored on disk. Most of the programs will accept data the user submits on punched cards or data stored previously on disk by the instructor. The latter option enables an instructor to assign problems involving large, realistic data sets without requiring his students to spend an inordinate amount of time keypunching.

SIMPLE LINEAR REGRESSION

Explanatory material has now been incorporated into the existing program for simple linear regression. When running this program, the student can request that a preliminary discussion be printed out and that the information necessary to follow-up discussions be stored on disk for three days. The preliminary discussion, appended to the usual output, describes the assumptions implicit in using linear regression techniques, explaining in detail the model assumed to be governing the experiment. The student can look over the output; if he does not understand what the numbers represent, he can request within the three-day limit that one or more of the follow-up discussions be produced.

A sample run, using data from Draper and Smith (1), is included at the end of this paper. The dependent variable is pounds of steam used monthly; the independent variable is average atmospheric temperature in degrees Fahrenheit. Only the plot of the regression line, confidence bands, and observed values is excluded.

The discussions are largely heuristic; we do not assume that the user is well versed in the theory of axiomatic probability and probability distributions or even in the usual statistical notation. The program requires that users give names to the dependent and independent variables, and these names, rather than Y and X, are used throughout the discussions. Probabilities and parameters are interpreted in terms of hypothetical repetitions of the experiment. Wherever possible, reference is made to the user's own data.

Most of the text of the discussion is the same for all users. Discussions of the estimated Y-intercept, the correlation coefficient, and the test of the hypothesis $\beta = 0$, however, vary somewhat with the user's results. The discussion of confidence intervals has also been made to vary with the sort of intervals calculated: the user may have requested that for each value of the independent variable in the data set, confidence intervals be printed for the estimated mean value of the dependent variable, for the predicted value of a future observation or for the predicted mean value of a number of future observations of the dependent variable.

The follow-up discussion entitled "Formulas" does not conform to the structure of the other discussions but was included to aid the student mystified by the calculations themselves.

We suspect that the discussions are most effective when the student is analyzing data he has collected from a field of application in which he is knowledgeable. However, if a class is composed of students all majoring, for instance, in the biological sciences, the instructor can assign a problem concerning biological data meaningful to the students and still take advantage of the program's disk data storage option.

TRANSPORTABILITY OF THE PROGRAM

This type of program makes modest demands on computer facilities. It is presently implemented on a computer having about 8000 sixteen-bit words of core storage, a line printer (with a 48-character character set) rated at 80 lines per minute, and a single disk drive. Such discussions should be quite easy to program on any comparable machine, and one should be able, in particular, to transfer the program shown here to other batch-mode computers having FORTRAN compilers without undue difficulty.

ACKNOWLEDGMENTS

The author is deeply indebted to Professor J. A. Warren for his invaluable suggestions, to Mr. Herbert Kirk for help in implementing the program, and to Professor Donald Guthrie for helpful comments.

REFERENCE

1. Draper, N. R. and H. Smith. Applied Regression Analysis. New York: John Wiley and Sons, 1966.

```
DATA FROM DRAPER AND SMITH, 'APPLIED REGRESSION ANALYSIS,' P. 8


            SIMPLE LINEAR REGRESSION


      MEAN X             MEAN Y             INTERCEPT A         SLOPE B
      0.52599E 02        0.94239E 01        0.1362298E 02      -0.7982868E-01


                         ANALYSIS OF VARIANCE

      SOURCE               DF                  SS            MEAN SQ.

      UNCORRECTED TOTAL    25       0.2284110E 04

      CORRECTION TERM       1       0.2220294E 04

      CORRECTED TOTAL      24       0.6381581E 02

      REGRESSION            1       0.4559239E 02      0.4559239E 02

      RESIDUAL             23       0.1822342E 02      0.7923226E 00

      R SQUARE=      0.7144372E 00

      CORRELATION COEFFICIENT =      -0.8452438E 00

      CORRECTED SS FOR X=      0.7154420E 04
                                     CONFIDENCE LIMITS FOR COEFFICIENTS

      COEFF          STANDARD ERROR    LOWER LIM.EST.     VALUE        UPPER LIM.EST.
      SLOPE            0.10523E-01    -0.10160E 00   -0.79828E-01    -0.58055E-01
      INTERCEPT        0.58146E 00     0.12419E 02    0.13622E 02     0.14826E 02
                                     CONFIDENCE LIMITS FOR Y

                              INTERVAL FOR PREDICTED MEAN VALUE OF Y(HAT)

    VALUE OF X        OBSERVED Y    LOWER LIM.EST.  ESTIMATED Y  UPPER LIM.EST.      RESIDUAL
       35.3000           10.9800          10.2782       10.8050         11.3318        0.1749
       29.7000           11.1300          10.6321       11.2520         11.8719       -0.1220
       30.8000           12.5100          10.5634       11.1642         11.7650        1.3457
       58.8000            8.4000           8.5367        8.9290          9.3213       -0.5290
       61.4000            9.2700           8.3063        8.7215          9.1366        0.5484
       71.3000            8.7300           7.3821        7.9312          8.4802        0.7987
       74.4000            6.3600           7.0829        7.6837          8.2845       -1.3237
       76.7000            8.5000           6.8590        7.5001          8.1412        0.9998
       70.7000            7.8200           7.4396        7.9791          8.5185       -0.1591
       57.5000            9.1400           8.6493        9.0328          9.4163        0.1071
       46.4000            8.2400           9.5266        9.9189         10.3112       -1.6789
       28.9000           12.1900          10.6819       11.3159         11.9499        0.8740
       28.1000           11.8800          10.7315       11.3795         12.0280        0.5001
       39.1000            9.5700          10.0304       10.5016         10.9729       -0.9316
       46.8000           10.9400           9.4976        9.8870         10.2763        1.0529
       48.5000            9.5800           9.3723        9.7512         10.1302       -0.1712
       53.3000           10.0900           8.4929        8.8891          9.2853        1.2008
       70.0000            8.1100           7.5065        8.0349          8.5633        0.0750
       70.0000            6.8300           7.5065        8.0349          8.5633       -1.2049
       74.5000            8.8800           7.0732        7.6757          8.2782        1.2042
       72.1000            7.6800           7.3052        7.8673          8.4294       -0.1873
       58.1000            8.4700           8.5976        8.9849          9.3722       -0.5149
       44.6000            8.8600           9.6551       10.0626         10.4700       -1.2026
       33.4000           10.3600          10.3995       10.9567         11.5138       -0.5967
       28.6000           11.0800          10.7005       11.3398         11.9792       -0.2598
                                                                                      -------
                                                                          TOTAL        0.0000
```

```
WHEN YOU USE THIS PROGRAM TO RELATE LBS STEAM USED/MO. TO ATMOS. TEMPERATURE,
YOU ARE CALCULATING A NUMBER OF VALUES CONSISTENT WITH THIS MODEL OF THE RELATIONSHIP --


LBS STEAM USED/MO. =  MU  +  BETA * (ATMOS. TEMPERATURE - AVERAGE ATMOS. TEMPERATURE)  +  EPSILON


FOR YOUR DATA SET, AVERAGE ATMOS. TEMPERATURE =         52.59999
                   MU IS ESTIMATED TO BE                 9.42399
                   BETA IS ESTIMATED TO BE              -0.07982


HENCE


'FITTED' VALUE OF LBS STEAM USED/MO. =         9.42399  +        -0.07982 * (ATMOS. TEMPERATURE -       52.59999)

                                LBS STEAM USED/MO. IS A RESPONSE VARIABLE.  THE MEASUREMENTS
                                OF LBS STEAM USED/MO. ARE INDEPENDENT, I.E., TAKING A
                                MEASUREMENT DOESN'T AFFECT THE VALUES OF OTHER MEASUREMENTS.
                                EACH RESPONSE VALUE, SUCH AS THE FIRST VALUE,     10.98000,
                                IS MADE UP OF THREE PARTS --


                                AN OVERALL MEAN OR AVERAGE RESPONSE,
                                A PART THAT DEPENDS ON THE VALUE OF ATMOS. TEMPERATURE, AND
                                A RANDOM ELEMENT.


THE OVERALL MEAN MU CAN BE THOUGHT OF AS THE AVERAGE VALUE
OF LBS STEAM USED/MO. THAT WOULD BE OBTAINED IF RESPONSES
WERE MEASURED INDEFINITELY OFTEN AT EACH OF THE VALUES OF
ATMOS. TEMPERATURE IN THIS DATA SET.

MU IS ESTIMATED BY THE 'MEAN LBS STEAM USED/MO.',

           9.42399

                                WHEN WE CALCULATE THE PART OF THE RESPONSE THAT DEPENDS ON
                                ATMOS. TEMPERATURE, WE ARE ASSUMING THIS --

                                IF MEASUREMENTS WERE MADE REPEATEDLY AT EACH VALUE OF
                                ATMOS. TEMPERATURE AND IF WE PLOTTED THE AVERAGE RESPONSE
                                FOR EACH VALUE OF ATMOS. TEMPERATURE AGAINST THE
                                CORRESPONDING VALUE OF ATMOS. TEMPERATURE, THEN
                                A LINE CONNECTING THE PLOTTED POINTS WOULD BE AT LEAST
                                APPROXIMATELY STRAIGHT.

                                IF THIS ASSUMPTION IS NOT TRUE, THEN THE CONTRIBUTION OF
                                ATMOS. TEMPERATURE TO THE VALUES OF LBS STEAM USED/MO. MAY
                                NOT BE PROPERLY REPRESENTED.

                                WHEN THIS ASSUMPTION IS TRUE, THEN THE CONTRIBUTION OF
                                ATMOS. TEMPERATURE TO LBS STEAM USED/MO. IS THE PRODUCT OF
                                THE SLOPE OF THAT LINE (BETA) AND THE VALUE OF THE DIFFERENCE,
                                ATMOS. TEMPERATURE -       52.59999.

                                FOR EXAMPLE, WHEN ATMOS. TEMPERATURE =        35.300,
                                THE CONTRIBUTION IS BETA * (     35.30000 -        52.59999)

                                BETA IS ESTIMATED BY 'SLOPE B',       -0.07982, THEREFORE
                                THE CONTRIBUTION WHEN ATMOS. TEMPERATURE =     35.30000
                                IS ESTIMATED TO BE       -0.07982 * (     35.30000 -        52.59999)
                                =        1.38103

                                                    THE RANDOM ELEMENT EPSILON EXPRESSES THE NOTION THAT IF ONE
                                                    MADE REPEATED MEASUREMENTS OF LBS STEAM USED/MO. AT A
                                                    SINGLE VALUE OF ATMOS. TEMPERATURE, SAY      35.30000,
                                                    THEN DIFFERENT RESPONSES WOULD BE OBTAINED.  THE RANDOM
                                                    ELEMENT REPRESENTS THE DIFFERENCE BETWEEN A SINGLE VALUE OF
                                                    LBS STEAM USED/MO. AND THE MEAN OF REPEATED OBSERVATIONS OF
                                                    LBS STEAM USED/MO. WHEN ATMOS. TEMPERATURE IS HELD CONSTANT.

                                                    THE RANDOM ELEMENT IN THE RESPONSE ASSOCIATED WITH ONE VALUE
                                                    OF ATMOS. TEMPERATURE, HOWEVFR, IS ASSUMED TO BE ABOUT AS
                                                    VARIABLE IN SIZE AS THE RANDOM ELEMENTS OF RESPONSES
                                                    ASSOCIATED WITH OTHER VALUES OF ATMOS. TEMPERATURE.
```

```
ANOTHER WAY OF WRITING THE MODEL ABOVE IS THIS --


LBS STEAM USED/MO. = ALPHA  +  BETA * ATMOS, TEMPERATURE  +  EPSILON


HERE ALPHA IS THE VALUE OF LBS STEAM USED/MO. AT
ATMOS. TEMPERATURE = 0 FOR THE STRAIGHT LINE CONNECTING
THE AVERAGE RESPONSES.

ALPHA IS ESTIMATED BY 'INTERCEPT A',      13.62298.
A   IS   AN EXTRAPOLATED VALUE, SINCE ATMOS. TEMPERATURE = 0
IS NOT INCLUDED IN THE RANGE OF ATMOS. TEMPERATURE.


                               IN ADDITION TO THE ESTIMATES DISCUSSED HERE, THE PROGRAM
                               DISPLAYS THE RESULTS OF OTHER CALCULATIONS WHICH CAN HELP
                               YOU IN ANALYZING YOUR DATA THOROUGHLY.  TO GET A DISCUSSION
                               OF A TOPIC LISTED BELOW, SUBMIT A FOLLOW-UP JOB WITH A 1
                               PUNCHED ON THE FIRST INPUT CARD IN THE COLUMN STIPULATED
                               UNDER THE TOPIC HEADING BELOW.  YOU ARE WELCOME TO REQUEST
                               MORE THAN ONE DISCUSSION THAT WAY IF YOU SO WISH.


* *  ANALYSIS OF VARIANCE TABLE AND R-SQUARED
     COLUMN 70

     THE ANALYSIS OF VARIANCE TABLE PRESENTS
INFORMATION ON THE VARIABILITY OF THE RESPONSES.


               * *  CORRELATION COEFFICIENT
                    COLUMN 71

                    A DESCRIPTIVE DEVICE WHEN THE MODEL IS
               AS STATED ABOVE, THE CORRELATION COEFFICIENT
               HAS A DIFFERENT STATISTICAL INTERPRETATION
               WHEN THE VALUES OF ATMOS. TEMPERATURE ARE NOT
               SET OR OTHERWISE FREE OF RANDOMNESS.


                               * *  CONFIDENCE INTERVALS AND STANDARD ERRORS
                                    COLUMN 72

                                    A CONFIDENCE INTERVAL PROVIDES A MEASURE
                               OF THE PRECISION OF AN ESTIMATE.


                                              * *  STATISTICAL TESTS OF HYPOTHESES
                                                   COLUMN 73

                                                   CONFIDENCE INTERVAL CALCULATIONS
                                              CAN BE USED TO PERFORM STATISTICAL TESTS.


                                                            * *  RESIDUALS
                                                                 COLUMN 74

                                                                 THE RESIDUALS ARE DEFINED TO BE THE
                                                            ACTUAL VALUES OF LBS STEAM USED/MO. MINUS THE
                                                            CORRESPONDING 'FITTED' VALUES OF
                                                            LBS STEAM USED/MO..  THEY ARE PARTICULARLY
                                                            USEFUL FOR DETECTING INADEQUACIES IN THE MODEL.


                                                                         * *  FORMULAS FOR QUANTITIES CALCULATED
                                                                              COLUMN 75

                                                                              THE MATHEMATICAL RELATIONSHIPS AMONG THE
                                                                         ESTIMATES, SUMS OF SQUARES, ETC. CAN BE SHOWN
                                                                         ALGEBRAICALLY.


REFERENCES --


DIXON AND MASSEY, 'INTRODUCTION TO STATISTICAL ANALYSIS, THIRD EDITION',
               MCGRAW-HILL, 1969.  PAGES 193 FF.

DRAPER AND SMITH, 'APPLIED REGRESSION ANALYSIS', JOHN WILEY AND SONS, 1966.
               PAGES 4 FF.

SNEDECOR AND COCHRAN, 'STATISTICAL METHODS, SIXTH EDITION', IOWA STATE
               UNIVERSITY PRESS, 1967.  PAGES 135 FF.

STEEL AND TORRIE, 'PRINCIPLES AND PROCEDURES OF STATISTICS', MCGRAW-HILL, 1960.
               PAGES 161 FF.
```

Follow-up Discussions

```
* *  ANALYSIS OF VARIANCE TABLE AND R-SQUARED

                        THE ANALYSIS OF VARIANCE TABLE PRESENTS INFORMATION ON THE
                        VARIABILITY OF THE RESPONSES.

THE UNCORRECTED TOTAL SUM OF SQUARES IS A CRUDE MEASURE OF HOW
MUCH THE RESPONSES VARY.  IT IS THE SUM OF THE SQUARED VALUES OF
LBS STEAM USED/MO..

              2                  2
   10.98000  + ... +    11.08000  =          2284.1101

                       THE UNCORRECTED TOTAL DOES NOT TAKE INTO ACCOUNT THE
                       LOCATION OF MOST OF THE VALUES OF LBS STEAM USED/MO..  THE
                       CORRECTION FOR THE MEAN OF LBS STEAM USED/MO. ALLOWS FOR
                       THE POSSIBILITY THAT THE RESPONSE VALUES MAY BE LARGE IN
                       SIZE BUT CLOSE TOGETHER ABOUT SOME LARGE CENTRAL VALUE.  IT
                       CAN BE EXPRESSED AS THE NUMBER OF OBSERVATIONS TIMES THE
                       SQUARE OF THE MEAN OF LBS STEAM USED/MO..

                                         2
                         25 * (       9.42399)  =        2220.2943

THE CORRECTED TOTAL SUM OF SQUARES IS THEN THE DIFFERENCE BETWEEN
THE UNCORRECTED TOTAL AND THE CORRECTION FOR THE MEAN.  A MEASURE
OF HOW MUCH THE RESPONSES VARY ABOUT A CENTRAL VALUE. IT CAN ALSO
BE EXPRESSED AS THE SUM OF THE SQUARED DIFFERENCES BETWEEN THE
ACTUAL VALUES OF LBS STEAM USED/MO. AND THE OVERALL MEAN OF
LBS STEAM USED/MO..

                                 2                                    2
(    10.98000 -       9.42399)  + ... + (     11.08000 -      9.42399)

=         63.8158

                        THE REGRESSION SUM OF SQUARES INDICATES HOW MUCH OF THE
                        VARIABILITY IN THE RESPONSES CAN BE 'EXPLAINED' BY THE
                        VARIABILITY IN THE VALUES OF ATMOS. TEMPERATURE.  IT CAN BE
                        EXPRESSED AS THE SUM OF THE SQUARED DIFFERENCES BETWEEN THE
                        FITTED VALUES OF LBS STEAM USED/MO. AND THE OVERALL MEAN OF
                        LBS STEAM USED/MO..

                                           2                                    2
           (     10.80503 -       9.42399)  + ... + (     11.33988 -      9.42399)

      =         45.5923

IF THE FITTED RESPONSE VALUES ARE CLOSE TO THE ACTUAL RESPONSE
VALUES. THEN THE REGRESSION SUM OF SQUARES WILL BE ALMOST AS BIG
AS THE CORRECTED TOTAL SUM OF SQUARES.  R-SQUARED, 0.71443,
IS THE RATIO OF THESE SUMS OF SQUARES.  IF R-SQUARED IS CLOSE TO
ONE, THEN THE MODEL AGREES WELL WITH THE DATA, AND THE RANDOM
ELEMENTS ARE LIKELY TO BE SMALL.  A LARGE R-SQUARED, HOWEVER,
DOES NOT GUARANTEE THAT THE MODEL IS THE CORRECT ONE, SINCE
R-SQUARED MAY BE LARGE FOR SEVERAL DIFFERENT MODELS ON THE SAME
DATA SET.
```

THE RESIDUAL SUM OF SQUARES IS THE DIFFERENCE BETWEEN THE CORRECTED TOTAL SUM OF SQUARES AND THE REGRESSION SUM OF SQUARES. IT IS ALSO THE SUM OF THE SQUARED DIFFERENCES BETWEEN THE ACTUAL VALUES OF LBS STEAM USED/MO. AND THE FITTED VALUES OF LBS STEAM USED/MO., THAT IS, THE SUM OF THE SQUARED RESIDUALS,

```
                               2                                     2
10.98000 -     10.80503) + ... + (    11.08000 -     11.33988)

     18.2234
```

THUS IT IS AN ESTIMATE OF THE VARIABILITY OF THE RANDOM ELEMENTS IF THE MODEL IS ACCURATE. IF THE MODEL IS POOR, ITS VALUE IS LIKELY TO BE INFLATED BY THAT 'LACK OF FIT'.

DEGREES OF FREEDOM ARE ASSOCIATED WITH EACH SUM OF SQUARES IN THE ANALYSIS OF VARIANCE TABLE. LOOSELY WE CAN SAY THAT ONE 'GAINS' A DEGREE OF FREEDOM WHEN ONE OBSERVES AN INDEPENDENT RESPONSE VALUE, AND ONE 'USES UP' A DEGREE OF FREEDOM WHEN ONE ESTIMATES A QUANTITY LIKE MU OR BETA.

THE UNCORRECTED TOTAL DEGREES OF FREEDOM ARE THUS THE NUMBER OF OBSERVATIONS, 25. THE CORRECTION FOR THE MEAN OF LBS STEAM USED/MO. USES UP ONE DEGREE OF FREEDOM, SINCE IT IS ASSOCIATED WITH THE ESTIMATION OF MU. THE CORRECTED TOTAL DEGREES OF FREEDOM ARE THEN 25 - 1 = 24.

OF THE CORRECTED TOTAL DEGREES OF FREEDOM, ONE IS USED UP BY THE REGRESSION SUM OF SQUARES, SINCE IT IS ASSOCIATED WITH THE ESTIMATION OF BETA. THE REMAINING DEGREES OF FREEDOM, 23, ARE ASSOCIATED WITH THE RESIDUAL SUM OF SQUARES.

THE MEAN SQUARES SHOWN ARE THE SUMS OF SQUARES DIVIDED BY THE DEGREES OF FREEDOM ASSOCIATED WITH THEM.

* * CORRELATION COEFFICIENT

THE CORRELATION COEFFICIENT, -0.8452, IS THE SQUARE ROOT OF R-SQUARED. IT IS POSITIVE WHEN B IS POSITIVE AND NEGATIVE (AS HERE) WHEN B IS NEGATIVE. LIKE R-SQUARED, IT IS A DESCRIPTIVE DEVICE WHEN THE MODEL IS AS STATED PREVIOUSLY.

HOWEVER, IF THE VALUES OF ATMOS. TEMPERATURE ARE NOT SET BY THE EXPERIMENTER OR OTHERWISE KNOWN TO BE EXACT, THEN THE CORRELATION COEFFICIENT HAS ANOTHER STATISTICAL INTERPRETATION. SUPPOSE EACH VALUE OF ATMOS. TEMPERATURE INCLUDES A RANDOM ELEMENT, AND THESE RANDOM ELEMENTS CAN BE DESCRIBED THE SAME WAY THAT WE DESCRIBED THE RANDOM ELEMENTS INCLUDED IN THE VALUES OF LBS STEAM USED/MO., THEN THE CORRELATION COEFFICIENT IS A MEASURE OF THE ASSOCIATION BETWEEN LBS STEAM USED/MO. AND ATMOS. TEMPERATURE.

IF THE CORRELATION COEFFICIENT IS CLOSE TO +1 OR -1, THE ASSOCIATION IS STRONG. A POSITIVE CORRELATION COEFFICIENT WOULD HAVE IMPLIED THAT LARGE VALUES OF LBS STEAM USED/MO. TEND TO APPEAR WITH LARGE VALUES OF ATMOS. TEMPERATURE. A NEGATIVE CORRELATION COEFFICIENT (LIKE -0.8452) IMPLIES THAT LARGE VALUES OF LBS STEAM USED/MO. TEND TO APPEAR WITH SMALL VALUES OF ATMOS. TEMPERATURE.

ONE CAN PERFORM A STATISTICAL TEST USING THE VALUE OF THE CORRELATION COEFFICIENT TO SEE, FOR EXAMPLE, IF THAT VALUE IS CONSISTENT WITH THE CLAIM THAT ATMOS. TEMPERATURE AND LBS STEAM USED/MO. ARE REALLY UNRELATED. SUCH A TEST IS DESCRIBED ON PAGE 190 OF STEEL AND TORRIE'S 'PRINCIPLES AND PROCEDURES OF STATISTICS'.

```
* *  CONFIDENCE INTERVALS AND STANDARD ERRORS

                    CONFIDENCE INTERVALS FOR SEVERAL DIFFERENT ESTIMATES ARE
                    PRINTED OUT BY THIS PROGRAM.  THE PROGRAM ALWAYS DISPLAYS
                    CONFIDENCE INTERVALS BASED ON 'SLOPE B', THE ESTIMATE OF
                    BETA, AND 'INTERCEPT A', THE ESTIMATE OF ALPHA.

THE STANDARD ERRORS OF A AND B ARE ALSO SHOWN.  THESE QUANTITIES
ARE ESTIMATES OF THE VARIABILITY OF A AND B WHICH ENTER INTO THE
CALCULATIONS OF THE RESPECTIVE CONFIDENCE INTERVALS.

                    WHEN YOU FIRST RAN THIS PROGRAM, YOUR SECOND CARD CARRIED A
                    CRITICAL VALUE OF THE STUDENT'S T DISTRIBUTION FOR LEVEL
                    0.050.  THEN ONE CAN SAY THAT IF RESPONSES WERE MEASURED OVER AND
                    OVER AGAIN FOR THE VALUES OF ATMOS. TEMPERATURE IN YOUR
                    DATA SET, 950 TIMES OUT OF 1000, THE CONFIDENCE INTERVAL
                    CALCULATED WOULD INCLUDE (I.E., 'COVER') THE TRUE VALUE OF
                    THE QUANTITY ESTIMATED.

FOR EXAMPLE, THE CONFIDENCE INTERVAL FOR 'SLOPE' (BETA) IS ALL
THE NUMBERS BETWEEN      -0.10160 (LABELED 'LOWER LIM. EST.')
AND      -0.05805 (LABELED 'UPPER LIM. EST.').  IF YOU SAY,
'THE TRUE VALUE OF BETA IS A VALUE BETWEEN THOSE LIMITS', YOU
WILL BE WRONG ONLY 5.0 PERCENT OF THE TIME YOU PERFORM THIS
PROCEDURE.  'SLOPE B', YOUR ESTIMATE OF BETA, IS THE CENTER OF
THE INTERVAL.

                    SIMILIAR STATEMENTS MAY BE MADE ABOUT THE CONFIDENCE INTERVAL
                    FOR ALPHA.  HOWEVER, SINCE A IS AN EXTRAPOLATED VALUE, IT
                    WOULD BE SAFER TO MAKE CONFIDENCE INTERVAL STATEMENTS ABOUT
                    THE MEAN OF LBS STEAM USED/MO. INSTEAD OF ALPHA.

WHEN YOU RAN  THIS PROGRAM, YOU CHOSE TO HAVE IT CALCULATE THE
LIMITS OF CONFIDENCE INTERVALS FOR
'PREDICTED MEAN VALUE OF LBS STEAM USED/MO.'
FOR EACH VALUE OF ATMOS. TEMPERATURE IN YOUR DATA SET.  FOR EACH
VALUE OF ATMOS. TEMPERATURE, THE CONFIDENCE LIMITS CAN BE
INTERPRETED AS FOLLOWS --


IF YOU WERE TO TAKE MEASUREMENTS OF
LBS STEAM USED/MO. OVER AND OVER AGAIN FOR THAT ATMOS. TEMPERATURE,
950 TIMES OUT OF 1000, A CONFIDENCE INTERVAL
CALCULATED THE WAY YOU HAVE DONE SO HERE WOULD
INCLUDE THE MEAN OF THOSE VALUES OF LBS STEAM USED/MO..

                    NOTE THAT THESE CONFIDENCE INTERVALS ARE BROADER FOR VALUES
                    OF ATMOS. TEMPERATURE FARTHER FROM THE MEAN
                    ATMOS. TEMPERATURE.  AS BEFORE, THE CONFIDENCE INTERVAL
                    IS THE NUMBERS BETWEEN 'LOWER LIM. EST.' AND 'UPPER LIM.
                    EST.'
```

```
* *  STATISTICAL TESTS OF HYPOTHESIS

                    THE CONFIDENCE INTERVAL CALCULATIONS CAN BE USED TO PERFORM
                    STATISTICAL TESTS.  IF YOU WANT TO TEST, SAY, THE HYPOTHESIS
                    THAT BETA HAS SOME SPECIFIC VALUE, YOU WOULD REJECT THE
                    HYPOTHESIS AT THE 0.050 LEVEL OF SIGNIFICANCE IF THAT
                    SPECIFIC VALUE IS NOT IN THE CONFIDENCE INTERVAL.  HOWEVER,
                    IF THAT SPECIFIC VALUE IS BETWEEN THE CONFIDENCE LIMITS, YOU
                    WOULD NOT REJECT THAT HYPOTHESIS.

FOR EXAMPLE, YOU MIGHT WANT TO TEST THE HYPOTHESIS THAT BETA = 0.
THIS HYPOTHESIS IS EQUIVALENT TO THE CLAIM THAT THERE IS NO
RELATIONSHIP (OF THE FORM OF THE MODEL, AT LEAST) BETWEEN
LBS STEAM USED/MO. AND ATMOS. TEMPERATURE.  IN YOUR DATA SET,
ZERO IS NOT BETWEEN         -0.10160 AND         -0.05805, SO YOU
WOULD     REJECT THE HYPOTHESIS THAT BETA = 0.


                    TESTS ON ALPHA AND THE 'FITTED' ('ESTIMATED') VALUES OF
                    LBS STEAM USED/MO. CAN BE MADE THE SAME WAY.  THE LEVEL OF
                    SIGNIFICANCE OF THE TESTS, 0.050, IS THE LEVEL OF THE
                    CONFIDENCE INTERVAL, WHICH YOU SPECIFIED ON YOUR SECOND CARD
                    WHEN YOU FIRST RAN THIS PROGRAM.

IT SHOULD BE NOTED THAT THE 'F' TEST OF THE REGRESSION -- SEEING
IF THE RATIO OF THE REGRESSION MEAN SQUARE TO THE RESIDUAL MEAN
SQUARE IS BIGGER THAN A CRITICAL VALUE OF THE APPROPRIATE F
DISTRIBUTION -- IS EQUIVALENT TO TESTING THE HYPOTHESIS THAT
BETA = 0 AS IT WAS DONE ABOVE.


  * *  RESIDUALS

                      A RESIDUAL IS DEFINED TO BE

    ACTUAL VALUE OF LBS STEAM USED/MO. - FITTED VALUE OF LBS STEAM USED/MO.

                      THE RESIDUALS ARE PARTICULARLY USEFUL FOR DETECTING
                      INADEQUACIES IN THE MODEL.  THEY REFLECT THE EFFECTS OF THE
                      RANDOM ELEMENTS IN THE RESPONSES IF THE MODEL IS A GOOD
                      DESCRIPTION OF THE DATA.  IF THE MODEL IS GOOD, THEN, THERE
                      SHOULD BE NO PARTICULAR PATTERN IN THE SIZES OF THE
                      RESIDUALS.  THE ONLY EXCEPTION IS THAT FITTED VALUES ARE
                      CALCULATED SO THAT THEIR SUM IS THE SUM OF THE ACTUAL VALUES
                      OF THE RESPONSES.  HENCE THE SUM OF THE RESIDUALS IS ALWAYS
                      ZERO.

  IF, HOWEVER, THE RESIDUALS ARE LARGE IN SIZE FOR LARGE VALUES OF
  ATMOS. TEMPERATURE AND SMALL FOR SMALL VALUES OF
  ATMOS. TEMPERATURE, ONE SHOULD BEGIN TO SUSPECT THAT THE EPSILONS
  DO NOT VARY IN THE SAME WAY FOR DIFFERENT ATMOS. TEMPERATURE, AS
  SUPPOSED.  IF THE RESIDUALS ARE NEGATIVE FOR VALUES OF
  ATMOS. TEMPERATURE FAR FROM THE MEAN ATMOS. TEMPERATURE AND
  POSITIVE FOR VALUES OF ATMOS. TEMPERATURE NEAR THE MEAN
  ATMOS. TEMPERATURE THEN ONE MIGHT SUSPECT THAT THE 'STRAIGHT
  LINE' ASSUMPTION OF THE MODEL IS UNTRUE, THAT IS, THE RESPONSE
  VALUES MAY DEPEND ON THE VALUES OF ATMOS. TEMPERATURE IN A
  DIFFERENT WAY.

                      THE TABLE BELOW LISTS THE OBSERVATIONS WHOSE RESIDUALS MAY
                      BE CONSIDERED UNUSUALLY LARGE.  ALTHOUGH RANDOM VARIATION
                      ALONE MAY ACCOUNT FOR THESE OUTLIERS, THEY SHOULD BE
                      EXAMINED CAREFULLY -- THE SIZE OF THE RESIDUALS MAY BE DUE
                      TO INCORRECT RECORDING OR KEYPUNCHING OF THE OBSERVATIONS,
                      FAULTY EXPERIMENTAL EQUIPMENT, ETC.  IN ADDITION OUTLIERS,
                      LIKE OVERALL PATTERNS IN THE RESIDUALS, MAY INDICATE A WAY
                      IN WHICH THE MODEL IS DEFICIENT.

  OBSERVED VALUE OF          OBSERVED VALUE OF
  ATMOS. TEMPERATURE         LBS STEAM USED/MO.          RESIDUAL
  ------------------         ------------------          --------



                      *** THERE ARE NO OUTLIERS ***
```

```
* *  FORMULAS FOR QUANTITIES CALCULATED

                    LET N = NUMBER OF OBSERVATIONS

                    LET X , X , ...., X  = VALUES OF ATMOS. TEMPERATURE
                         1   2         N
                    LET Y , Y , ...., Y  = VALUES OF LBS STEAM USED/MO.
                         1   2         N
                    THEN THE FOLLOWING FORMULAS SHOW HOW THE VARIOUS ESTIMATES,
                    ETC. ARE CALCULATED.

'MEAN X' = (X  + X  + ... + X ) / N
             1    2          N

         = (    35.30000 + ... +     28.60000) /   25

         =       52.59999

'MEAN Y' = (Y  + Y  + ... + Y ) / N
             1    2          N
         = (    10.98000 + ... +     11.08000) /   25

         =        9.42399
                                 2                2                2                      2
'CORRECTED SS FOR X' = S   = (X   - MEAN X)  + (X   - MEAN X)  + ... + (X   - MEAN X)
                        X      1                2                        N

                                               2                                          2
         = (    35.30000 -       52.59999)  + ... + (    28.60000 -        52.59999)

         =       7154.42047

                (X  - MEAN X) * (Y  - MEAN Y)  + ... +  (X  - MEAN X) * (Y  - MEAN Y'
                  1               1                       N               N
'SLOPE B' = B = -------------------------------------------------------------------------
                                                   2
                                                  S
                                                   X

          ( (    35.30000 -     52.59999) * (    10.98000 -       9.42399) + ...  )
         = -----------------------------------------------------------------------
                                                 7154.42047

         =       -0.07982

'INTERCEPT A' = A = MEAN Y  -  B * MEAN X

         =       9.42399 - (       -0.07982 *        52.5999y)
         =       13.62298

                                                     TH,
LET Y(HAT)  = A + B*X , AND LET E  DENOTE THE I        RESIDUAL, Y  - A - B*X
          I              I        I                              I            I

= Y  - Y(HAT) ,  THEN
   I         I

'UNCORRECTED TOTAL SS' = UNCORRECTED TOTAL SUM OF SQUARES

             2             2
          = Y  + ... + Y
             1             N

                     2                       2
         =    10.98000  + ... +     11.08000  =          2284.1101
```

```
                                                                  2
'CORRECTION TERM' = CORRECTION FOR MEAN OF Y = N * (MEAN Y)

                                2
        =   25 * (        9.42399)  =           2220.2943

'CORRECTED TOTAL SS' = CORRECTED TOTAL SUM OF SQUARES

           = UNCORRECTED TOTAL SS - CORRECTION TERM

                       2                          2
           = (Y  - MEAN Y)  + ... + (Y  - MEAN Y)
               1                      N

                                          2                                        2
           = (     10.98000 -         9.42399)  + ... + (     11.08000 -         9.42399)

           =           63.8158

'REGRESSION SS' = SUM OF SQUARES DUE TO REGRESSION

                                     2                                  2
               = ( Y(HAT)  - MEAN Y )  + ... + ( Y(HAT)  - MEAN Y )
                         1                             N
                  2     2
               = B  * S
                        X
                                          2                                        2
           = (     10.80803 -         9.42399)  + ... + (     11.33988 -         9.42399)

           =           45.5925

'RESIDUAL SUM OF SQUARES' = SUM OF SQUARES ABOUT REGRESSION

                        = CORRECTED TOTAL SS  -  REGRESSION  SS

                                          2                           2
                        = (Y  - Y(HAT) )  + ... + (Y  - Y(HAT) )
                            1         1             N         N
                             2              2
                        = E  + ... +  E
                            1             N
                                         2                                     2
           = (     10.98000 -      10.80503)  + ... + (     11.08080 -     11.33988)

           =           18.2234

'UNCORRECTED TOTAL DF' = TOTAL DEGREES OF FREEDOM = N =    25

'CORRECTION TERM DF' = DEGREES OF FREEDOM FOR CORRECTION FOR MEAN Y = 1

'CORRECTED TOTAL DF' = CORRECTED TOTAL DEGREES OF FREEDOM = N - 1 =     24

'REGRESSION DF' = REGRESSION DEGREES OF FREEDOM = 1

'RESIDUAL DF' = RESIDUAL DEGREES OF FREEDOM = N - 2 =     23

'REGRESSION MEAN SQ.' = REGRESSION MEAN SQUARE = REGRESSION SS

                            2
'RESIDUAL MEAN SQ.' = S  = RESIDUAL SS / (N - 2)

                    18.2234 /    23 =          0.79232
```

```
 2
R  = REGRESSION SS / CORRECTED TOTAL SS =  0.71443

                                                                 1/2
'CORRELATION COEFFICIENT' = B * S  /  (CORRECTED TOTAL SS)
                                 X

                                                1/2                    1/2
        =       -0.07982 * (      7154.42047)     / (        63.81581)

        =             -0.84524

LET T DENOTE THE DESIRED PERCENTAGE POINT OF THE STUDENT'S T DISTRIBUTION WITH
N-2 DEGREES OF FREEDOM.  THEN CONFIDENCE INTERVALS MAY BE REPRESENTED THUS --

QUANTITY ESTIMATEC      ESTIMATE    DISTANCE FROM ESTIMATE TO LIMIT OF INTERVAL
------------------      --------    -------------------------------------------

RETA                    B           T * S / S
                                             X

                                              2           2 1/2
                                    T * S * (X    +...+ X    )
                                              1          N
ALPHA                   A           ---------------------------

                                           1/2
                                          N     * S
                                                   X

PREDICTED 'MEAN'        Y(HAT)
                             0                                      2   2  1/2
VALUE OF Y FOR A                    T * S * ( 1/N   +   (X   - MEAN X) /S     )
                        =A+B*X                            0               X
GIVEN X                      0
       0

PREDICTED VALUE OF
                                                                  2   2  1/2
A NEW OBSERVATION       Y(HAT)      T * S * ( 1  +  1/N  +  (X   - MEAN X) /S     )
                             0                                0                X
OF Y FOR A GIVEN X
                  0

PREDICTED VALUE OF

MEAN OF M NEW                                                       2    2    1/2
                        Y(HAT)      T * S * (1/M  +  1/N  +  (X   - MEAN X) /S     )
OBSERVATIONS OF Y            0                                 0                X

FOR A GIVEN X
             0

PREDICTED VALUE OF                                             2    2 1/2
                        X(HAT)      T * S * (C  +  C/N  +  X(HAT)  /S   )  / (B*C)
X FOR A GIVEN Y              0                                 0    X
               0
                        =(Y -A)/B               2     2       2      2
                           0        WHERE C = 1 - ( T  * S  / (B  * S  ) )
                                                                     X

FOR EXAMPLE, YOU SPECIFIED T =  2.06900.  THEREFORE, A 95.0 PERCENT CONFIDENCE
INTERVAL FOR BETA WOULD BE THE INTERVAL FROM

                                                1/2                   1/2
       -C.07982  -  2.06900 * (          0.7923)    / (       7154.4204)        TO

                                                1/2                   1/2
       -0.07982  +  2.06900 * (          0.7923)    / (       7154.4204)

I.E.,FROM     -0.10160 TO     -0.05805.
```

STATLAB, A SIMPLE PROGRAMMING SYSTEM FOR THE STATISTICS LABORATORY

G. F. Atkinson
University of Manitoba
Winnipeg, Canada

INTRODUCTION

In a course in statistical methods experience in analyzing real data is generally considered essential but such analyses, if they even approximate real problems, require considerable tedious calculation. To relieve this tedium, it is natural today to turn to the talents of the computer. This has been done by providing packaged programs which perform the required analyses, by requiring knowledge of a programming language as a prerequisite, or by teaching elementary programming as part of the Statistics course.

None of these alternatives appeared desirable to us. As consultants we had seen too many horrible examples of reliance on packaged programs to make us want to encourage their use; a programming prerequisite was felt to place a serious limitation on the students eligible for the course; and we did not wish to take time from the statistical material to teach a worthwhile amount of programming.

Our requirement then was for a language which was easy to learn and which could conveniently perform the calculations required for the statistical procedures covered by our course.

During the session 1969-70 the OMNITAB system was introduced into our Statistics labs. Our experience was generally favorable but for our purposes we found that OMNITAB has several disadvantages. OMNITAB is a very general system and as such is relatively large, relatively slow, and consequently expensive. OMNITAB was not designed for statistical calculations so a certain degree of "trickery" was necessary in many analyses. We also found that the students had more difficulty in mastering the language than had been anticipated.

Our experience, however, was favorable enough that it was decided to produce an easy-to-learn, easy-to-use programming system specifically designed for the problems to be solved in the statistical methods labs.

Basic Features of STATLAB

In the design of the system that was eventually named STATLAB, the following criteria were established:

1. Operations on whole arrays and vectors would form the basis of the system.
2. The syntax of the language would be extremely simple.
3. Dimensions of arrays and vectors would, as far as possible, be under the control of the system.
4. User written loops would not be permitted.
5. Input of statements and of data would be in free format, output would be in fixed format under system control.
6. A complete set of matrix operations would be included.

It was further decided that the system would be interpretive; that is, statements are interpreted and executed as they are encountered.

For our purpose it was considered essential that the system be capable of performing the calculations for the following type-problems:

1. One- and two-way analysis of variance.
2. Multiple linear regression.
3. One- and two-way analysis of covariance.
4. Chi-square in two-way contingency tables.

With a little ingenuity it is possible to do analysis of variance or covariance in a three-way classification, polynomial regression, and some other complex analysis.

In the STATLAB system variables may be scalars, vectors, or arrays; and operations are on whole vectors and arrays. To facilitate analysis of variance calculations function-type operators are included which obtain the row and column sums of an array giving a vector result and which obtain the sum and sum of squares (uncorrected) of all elements of array or vector, the result being a scalar. These operators together with arithmetic operations permit easy calculation of the sums of squares and mean squares required for the analysis.

To simplify references within the system a strict naming convention was established. Variable names consist of a letter (S, V, or A for scalars, vectors, and arrays respectively) plus a numeric index to uniquely identify the variable. These indices are limited to the range 1 to 25 for scalars, 1 to 20 for vectors, and 1 to 10 for arrays; so variable names appear as S1, S15, V2, V5, A7, etc. Arithmetic expressions consist of the result and two operands, e.g., S1 = S2 - S3. Function type statements, in general, consist only of the result, function-name and operand, e.g., S1 = TOTAL A1. The equal sign in these expressions has the usual meaning "is assigned the value", thus the statement

S2 = SUMSQR A1

means "Calculate the sum of the squares of all elements of the array A1 and assign this value to scalar variable S2". The dimensions of arrays and vectors, once established, are stored by the system and referenced as required.

Sample Program

A simple example will illustrate how STATLAB is used. Suppose one wishes to calculate the mean and variance of a sample of 7 observations:

```
1.2     3.4     5.1     6.2     1.3     4.5     2.0
```

A program to do this might be as follows:

```
STATLAB MEAN AND VARIANCE EXAMPLE
READ V1 7
1.2    3.4    5.1    6.2    1.3    4.5    2
S1 = TOTAL  V1          SUM OF OBSERVATIONS
S2 = S1 * S1
S2 = S2/7               CORRECTION FACTOR
S3 = SUMSQR VI          SUM THE SQUARES
S3 = S3 - S2            SUM OF SQUARES
S3 = S3/6               VARIANCE
S1 = S1/7               MEAN
PRINT S1  S3
STOP
```

In the first statement the word STATLAB signals the beginning of a program, the other information is for identifying purposes and is optional. The next statement instructs the program to READ 7 numbers from the following card or cards and to store the data as a vector with symbolic name V1. This statement defines the variable V1 and establishes its dimensions for future use. Next the sum (TOTAL) of the observations is obtained and stored as a scalar with name S1. Comments are freely used to identify the different steps and the calculations proceed until the statement STOP identifies the end of the program.

While it may appear that the restricted form of expressions might be undesirable, our experience is that this step-by-step approach is easy for the students to grasp and in fact forces their attention onto the procedures being used.

STATLAB was used with our third year general course in statistical methods which is taught at the level of Snedecor and Cochran. During the past year there were 80 students in the class (about 30% graduate students from other Departments). A lab assignment was given every week; student demonstrators were available 6 hours a week to provide any needed assistance but the students were not required to attend. One regular lecture period was devoted to an introduction of the basic features of the language. More complicated techniques were introduced as they were required but this seldom amounted to more than a five-minute digression from the regular lecture material.

Student acceptance of the system has been good, many had no previous exposure to computers and STATLAB has proven to be a relatively painless introduction. Because no detailed accounting of student jobs is done by the Computer Center's system, no very precise cost figures can be given but costs were estimated to be less than 25 cents per student job. The more careful students are frequently able to obtain an error free program on the first run.

Some initial difficulties were encountered by students who were simultaneously taking a FORTRAN course because of the marked similarities of the statements but the more restrictive rules of STATLAB. Some graduate students in the course have gone beyond our original intention of providing only a teaching tool by using the system for data analysis in their thesis problems (and occasionally in other courses).

The STATLAB Language

In this summary of the language and principles of operations the following notation will be used:

Variables will be represented by the type letter (S, V, or A) plus a symbolic index i, j, or k.
Constants will be represented by XXXX.
Dimensions will be represented by n, m, r, or c.

Input Statements

```
READ Si XXXX
READ Vi n
READ Ai r c
```

To "Read" a scalar the value to be assigned to the variable Si is punched as a constant in the statement card. In the form to read a vector or array the dimensions are punched in the statement card; data is punched on cards following the READ statement in free format with individual items separated by blanks or commas. Arrays are read in row order.

Output Statements

```
PRINT Si ... Sk  (up to 10 operands)
PRINT Vi
PRINT Ai
```

Variables printed are identified by their name and are printed in FORTRAN E-format. Vectors and arrays are printed with 5 elements per line.

Arithmetic Statements

Addition, subtraction, multiplication, and division are represented by the symbols +, -, *, and /. If the symbol @ represents <u>any</u> of these arithmetic operations, then the statements permitted may be summarized as:

```
Si = Sj @ Sk
Vi = Vj @ Vk
Ai = Aj @ Ak
Vi = Vj @ Sk
Ai = Aj @ Sk
Si = Sj @ XXXX
Vi = Vj @ XXXX
Ai = Aj @ XXXX
```

For vectors and arrays, arithmetic operations are element by element. Thus Aj * Ak means "multiply corresponding elements" and is <u>not</u> matrix multiplication. Where the first operand is a vector or array and the second a scalar or constant, the operation is on each element of the vector or array.

Matrix Multiplication

The letter M is used in the system to denote matrix multiplication; the allowed forms are:

```
Si = Vj M Vk     inner product of vectors
Ai = Vj M Vk     column vector by row vector
Vi = Vj M Ak     row vector by matrix
```

Vi = Aj M Vk   matrix by column vector
Ai = Aj M Ak   multiplication of matrices

In all these operations the dimensions of the operands are automatically checked to ensure conformability.

Function Statements

Most of the function type statements have vectors or arrays as operands. The most commonly used of these statements are:

Si = TOTAL Vj
Si = TOTAL Aj
Si = SUMSQR Vj
Si = SUMSQR Aj

which obtain the sum and sum of squares of all elements in a vector or array.

Vi = ROWSUM Aj
Vi = COLSUM Aj

calculate the sum of elements in the rows or the columns of an array and store the results as a vector.

Also of the function type are statements:

Ai = TRANSPOSE Aj
Ai = INVERSE Aj

Function statements are also included to obtain square roots, logs, and sums of products. To provide flexibility in handling data, special functions are provided to extract elements from arrays or vectors (and sub-vectors from vectors and sub-arrays from arrays). There is also limited facilities for combining variables to form vectors and arrays.

Control Statements

The "word" STATLAB signals the beginning of an individual program and the word STOP the end of a program. Programs can be batched and these statements provide for program separation and initialization. ERASE Vi and ERASE Ai are used to eliminate unwanted variables and thus free storage for other use.

Principles of Operation

In the STATLAB system vectors and arrays are stored dynamically in an array of 5000 words of storage. An associated array contains, for each vector or array, a pointer to the location in the storage array and dimensions. Scalars are stored statically in a vector of 25 words.

At the beginning of a program all variables are considered to be undefined. A variable is defined by its appearance as the operand of a READ statement or as the result of an operation (arithmetic or functional). Undefined variables cannot be used as operands in any statement except a READ.

The main program of the system is written in 360 Assembler Language. This portion interprets the statements, converts data, and calls FORTRAN routines which perform the actual operations. During the interpretation phase extensive syntax checking is performed, checking continues during execution (e.g., for conformability). If an error is detected a message is printed and further execution is terminated, interpretation and syntax checking continue.

Compiled under FORTRAN H the program requires slightly less than 70K of core. A typical student job (two-way analysis of variance with 4 rows, 5 columns and 3 observations per cell) requires less than 0.5 seconds of CPU time and produces about three pages of output.

Implementation and Future Plans

The University of Manitoba Computer Centre operates an IBM 360/65 with the usual complement of disks, tapes, etc. In the summer of 1970 a 2501 Card Reader with an Optical Mark Reader and a 1403 Printer was installed as a Student (or Debug) Terminal. This terminal with its 130K of dedicated core makes available STATLAB, WATFIV, SPECTRE, ALGOLW, PL/C, TUTOR, and FACET with a turnaround time of about one minute. The facility has proven very popular with the students, approximately 20,000 jobs per week were run during the period October-December.

STATLAB (and the other programs) can also be used from another terminal on campus which operated in a conventional batch mode and without the OMR reader but most students preferred to use the Student Terminal because of its "instant" turnaround and the convenience of the OMR feature.

Our experience with STATLAB operating under the "instant batch" processor has been quite satisfactory and no fundamental changes in the system are planned. A plotting routine is to be added to the system and some modifications to make multiple regression somewhat easier are being considered. A more straightforward method for doing analysis of variance in higher way classifications (and in complex nested designs) would be desirable but is difficult to include in the present system.

Acknowledgements

The author wishes to acknowledge the valuable contribution of Mr. Roy Kury who did much of the detailed programming and testing and who assisted greatly in the evolution of the system. Mr. Gordon Armstrong and many other members of the staff of the Computer Centre, University of Manitoba, were patient and helpful allies at many sticky points. Last, but not least, financial support from the Dean of Science was an essential factor in the development of the system.

Appendix: STATLAB Statements

The following is a complete list of the allowed STATLAB Statements:

Variables are represented by the type letter plus symbolic index i, j, k.

Constants are represented by XXXX.

Integers are represented by n, m, r, or c.

The symbol @ represents one of the arithmetic operations +, -, *, or /.

Control Statements

STATLAB
STOP
NOTE
ERASE Vi
ERASE Ai

Input-Output Statements

READ Si XXXX
READ Vi n
READ Ai r c
PRINT Si ... Sj (up to 10 operands)
PRINT Vi
PRINT Ai

Arithmetic Statements

All operations are element by element.

```
Si = Sj @ Sk               Si = Sj @ XXXX
Vi = Vj @ Sk               Vi = Vj @ XXXX
Ai = Aj @ Sk               Ai = Aj @ XXXX

Vi = Vj @ Vk
Ai = Aj @ Ak
```

Matrix Multiplication

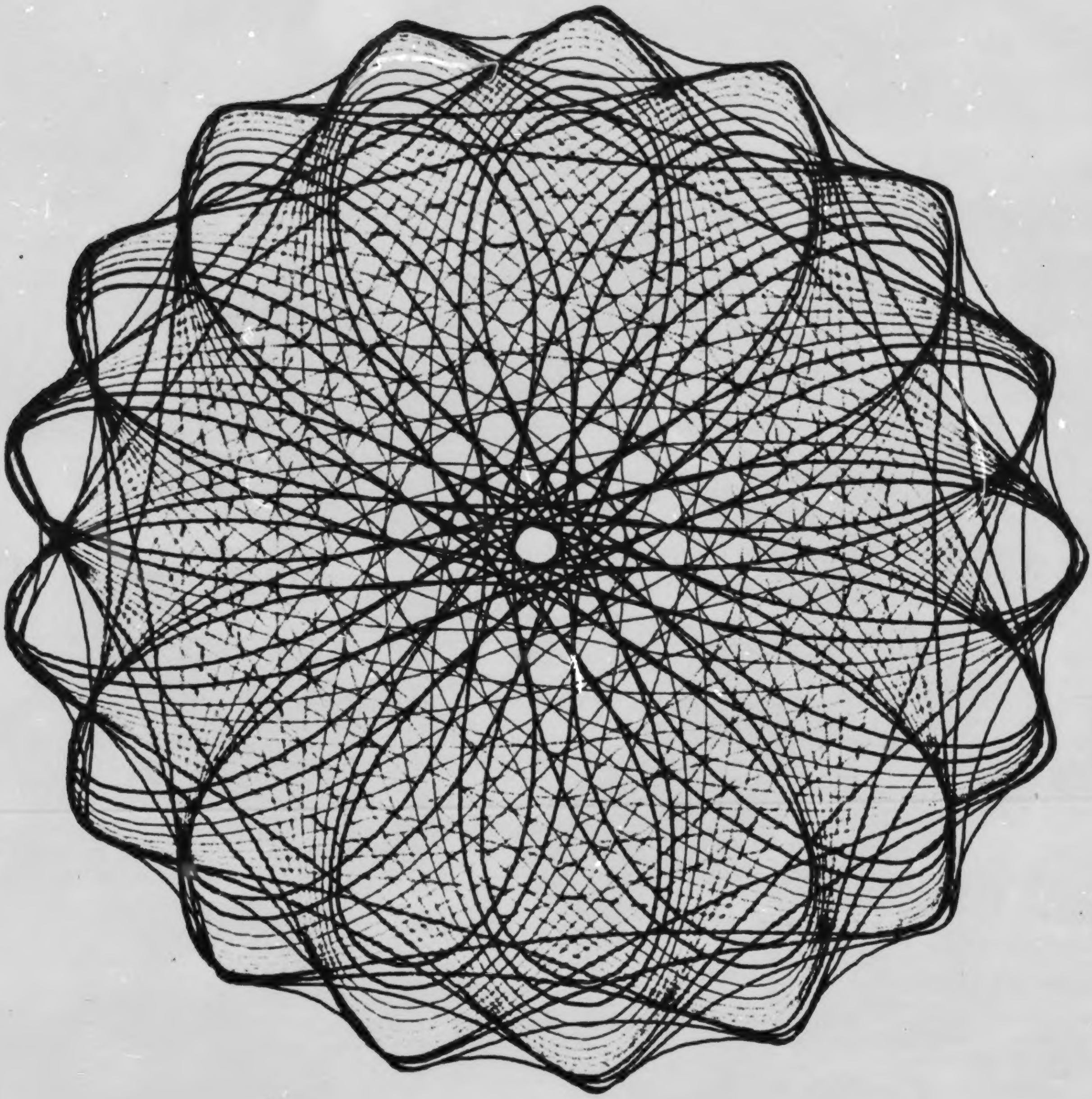
```
Si = Vj M Vk
Ai = Vj M Vk
Vi = Vj M Ak
Vi = Aj M Vk
Ai = Aj M Ak
```

Powers

```
Si = Sj P Sk               Si = Sj P XXXX
Vi = Vj P Sk               Vi = Vj P XXXX
Ai = Ai P Sk               Ai = Aj P XXXX
```

Function Statements

```
Si = TOTAL Vj              Si = TOTAL Aj
Si = MEAN Vj               Si = MEAN Aj
Si = SUMSQR Vj             Si = SUMSQR Aj

Vi = ROWSUM Aj             Vi = COLSUM Aj

Ai = TRANSPOSE Aj          Ai = INVERSE Aj

Si = SQRT Sj               Si = LOG Sj
Vi = SQRT Vj               Vi = LOG Vj
Ai = SQRT Aj               Ai = LOG Aj

Si = SUMPROD Vj Vk         Si = SUMPROD Aj Ak

Ai = ARRAY Vj r c

Si = EXTRACT Vj n          Si = EXTRACT Aj r c
Vi = EXTRACT Vj n m        Ai = EXTRACT Aj r c n m

Vi = COMPOSE Sj ... Sk
Vi = COMPOSE Vj ... Vk

Vi = DECOMPOSE Sj ... Sk
Ai = DECOMPOSE Vj ... Vk

Vi = DECX n Sj ... Sk
Vi = DECX c Vj ... Vk
```



Robert Spitz

EXPERIENCE WITH A PLATO MECHANICS COURSE

Bruce Sherwood, Carol Bennett,
Joan Mitchell, and Charlene Tenczar
University of Illinois
Urbana, Illinois 61801

ABSTRACT

The PLATO computer-based education system is being used to tutor students in an introductory mechanics course. The rationale for using the computer is discussed. Preliminary results are given on the effect the computer has had on the course; further results will be available at the time of the conference. An example of one of the tutorial lessons is given in an Appendix.

Physics lessons are being developed for the PLATO (1) computer-based education system at the University of Illinois. Most of these lessons are being used in a classical mechanics course for science and engineering freshmen. The initial motivation for writing these computer-based lessons was to improve certain aspects of teaching physics which are difficult to handle well in a large lecture, in a class of twenty students, or with a textbook. Among these aspects are one-to-one tutoring, graphical or pictorial aids to visualization, and computational aids. The original intention was that PLATO would supplement and enrich the ordinary classroom instruction. Initial testing has indicated that the computer can play a more important role in assuring mastery of topics at each stage of the course, and the computer activity is increasingly considered essential rather than merely supplemental.

In order to understand the essential aspect of the computer's role in the course it is important to review some of the problems associated with introductory physics courses.

There are many weaknesses in the organization of the typical introductory physics course. The traditional structure assumes that the student will read the assigned textbook chapter, get additional insight from a lecture, work the assigned homework problems, and finally discuss these problems in a small recitation class. In fact, the typical student does not adequately read the text, gets lost or bored in the lecture due to lack of preparation (or because the pace is too fast), and is ill-equipped to do the homework problems. For example, a poll of one hundred students in one of the introductory courses for science and engineering students at the University of Illinois indicated that, for a typical assignment of five homework problems, thirty percent of the students attempted none of these problems before coming to the recitation class. An additional ten percent of the students failed to master even one problem before class. No wonder the average student does not participate in the related class discussion. Yet the class discussion is the last formal instruction in the topic. Quizzes are of little assistance except for determining final grades since the results aren't known for several days. By this time the course has moved on to new material whetner or not the previous topic was understood.

Lack of mastery of one topic usually hinders the learning of the next topic. Looked at in this light it is not surprising that the average student finishes the course without a thorough understanding of any topic. This is reflected in the broad Gaussian distribution of examination scores, often centered at 50% correct: most students only half-understand the material. This state of affairs is not acceptable.

One attractive way to break this vicious cycle is through a self-paced course such as the one at MIT (2). The student is given a study guide to readings and related problems for a particular topic. He works independently at his own pace on this topic until he feels he has mastered it, at which time he takes a quiz. The quiz is graded immediately by an undergraduate grader. If he does very well on the quiz, he is given the study guide for the next topic; otherwise he must continue studying until he can perform well on a similar quiz. This procedure assures a high level of mastery of the subject material at each stage of the course. This brief description of the self-paced method omits many important features. The interested reader is referred to the MIT report (2).

The computer offers alternative means for emphasizing mastery. This role for the computer is presently (Spring 1971) being tested in a freshman mechanics class of eleven volunteer students. The course is divided into weekly topics. At the beginning of the week the students work through a two-hour tutorial computer lesson which presents the basic concepts, tutors the student on sample applications, and administers and immediately grades a quiz. The student must master the quiz before going on to other material. (This is the ideal: at the present time only a few of the computer lessons have reached this level of development.) This initial experience with the topic prepares the student well for further study and class discussion.

A one-hour discussion period immediately follows the computer work, and there is another at the end of the week after there has been time for thought and homework. These discussion periods have been unusually fruitful, presumably because each student has achieved a rather high level of understanding before coming to class. The students participate actively in the discussion, answering questions in areas they have mastered and asking well-formulated questions about areas of confusion.

In a regular discussion class most students are passive and avoid interaction with the instructor because of their lack of preparation: they cannot even ask for help on obscure points. In such a situation it is difficult for even the best instructors to accomplish much. On the other hand, with a class which has worked through a tutorial computer lesson an instructor can build creatively on a solid base of common understanding.

Similar preparation could theoretically be achieved from textbook study. A very strong student might ask himself questions, work out applications of the concepts, and quiz himself as he reads his textbook, but this student represents a tiny minority. Most students read their textbook passively. A programmed textbook can raise the level of interaction compared with an ordinary book, provided that the student does not simply look at the answers and read passively. But a programmed textbook would be impossibly complicated and clumsy if it attempted to approximate the intensity of interaction of a computer-based lesson and included safeguards against passive skimming. An example of such a computer-based lesson is included as an Appendix; additional lessons will be illustrated with slides at the conference (3).

The computer's essential contribution is that it forces the student into an active role. The student must continually respond to questions about the material. He is forced to review the material if he doesn't perform well on a quiz. The highly interactive nature of this work contrasts sharply with the student's passive role when reading a textbook or listening to a lecture. This activity makes possible his active participation in later discussion sessions.

Eventually this higher level of activity should be reflected in improved examination scores. Exam results from the small course now in progress will be available by the time of the conference. The most noticeable result of the experiment so far is the greatly improved participation in class discussions as compared with previous experience in similar courses and as compared with discussion sessions in those weeks where no computer lesson was yet available.

It should be emphasized that the role of the instructor becomes more important and more interesting with the introduction of computer tutoring. During the computer session most of the instructional load is carried by the computer lesson, freeing the instructor to treat individual questions as they arise. In the discussion sessions he need not dwell on the simplest concepts but is freed to concentrate on more challenging matters.

The mastery of the basic concepts is the bread-and-butter issue in improving physics education. But the computer can do more: it can simulate complex phenomena and offer the student the opportunity to write his own programs to solve difficult problems. These "simulation" and "calculational" uses of the computer have been widely developed in physics education (4). From our experience it appears that these modes can provide significant enrichment but cannot by themselves assure mastery of the basic material. When the students played with simulation material, guided only by printed notes with suggestions of things to try, the discussion periods that followed were much less stimulating and useful than after tutorial lesson study.

Most of the PLATO mechanics lessons involve simulation, including graphical visualization, to aid in the tutoring process. (See the Appendix for an example.) The usefulness of simulations appears to be considerably expanded when imbedded in a tutorial framework. As to the computational or programming mode, students who complete their basic work early are encouraged to write programs to solve problems related to the lesson material. An important aspect of the power of the PLATO system and the TUTOR (5) author language is that these various modes of computer use can be closely integrated.

REFERENCES

1. D. Bitzer, R. Blomme, B. Sherwood, and P. Tenczar, "The PLATO System and Science Education", in Proceedings of a Conference on Computers in Undergraduate Science Education, Illinois Institute of Technology and Commission on College Physics, August 1970 (to be published). D. Alpert and D. Bitzer, "Advances in Computer-Based Education", Science 167, 1582 (1970).

2. B. A. Green, Jr., "A Self-Paced Course in Freshman Physics", Massachusetts Institute of Technology Education Research Center (1969).

3. For brief descriptions of several other mechanics lessons, see B. Sherwood, C. Bennett, R. Bucher, L. Cannell, J. Mitchell, and C. Tenczar (to be published). A detailed description of a lesson on dynamics problems involving free-body diagrams is given by B. Sherwood (to be published).

4. For example, see the Proceedings of a Conference on Computers in Undergraduate Science Education, Illinois Institute of Technology and Commission on College Physics, August 1970 (to be published).

5. R. A. Avner and P. Tenczar, "The TUTOR Manual", Report X-4 of the Computer-Based Education Research Laboratory (1969).

APPENDIX: A PLATO TUTORIAL LESSON ON MOMENTUM CONSERVATION

One of the lessons used in the computer-based mechanics course deals with momentum conservation in collisions. This lesson will be briefly described in order to illustrate some of the techniques used in tutoring the student and testing for mastery.

The accompanying figures are photographs of one student's television display screen. There are twenty such display consoles in the PLATO III system. Each student works at his own rate and is treated individually. Each student has a silent typewriter keyboard which has all the standard keys plus some control keys.

Figure 1 shows an index page to which the student may go at any time. The main subdivisions of the lesson are listed, and the student may choose any section he has previously studied. (In this lesson he is not allowed to skip over material.) The main purpose of this index is to permit the student to review pertinent sections if his performance on the quiz is not adequate.

Figure 2 shows an example of a tutorial question. The student answered incorrectly. He may try again or he may press the HELP key for further discussion. Figure 3 shows the help sequence beginning with the first of several questions. The momentum vector diagram on the right is filled in as the student answers the questions correctly.

Figure 4 shows a later step in the lesson. In this case the student observes an animated simulation of a collision to get necessary information. Figure 5 is the end result of this simulation: included are numerical data on initial and final momenta. The student uses these data in order to answer the questions posed in Figure 4.

Figures 6 and 7 show two of five quiz questions. The student has only one chance at answering each question and is given no help. He must get all five questions right before he can proceed to additional material. Otherwise he must review and try the quiz again, which involves going through all the questions again. The questions are presented in random order, and the numerical ones contain randomly chosen parameters. On the average, students had to take the quiz twice. Students complained about having to repeat questions which they had previously answered correctly, particularly since some involve quite a bit of arithmetic. It is possible that repeating only the previously-missed questions might lead to undesirable effects such as inadequate integration of the various concepts. A reasonable compromise might be to repeat some questions previously answered correctly, in addition to those previously missed.

In Figure 8 the simulated collision display is augmented on the right by a view of the collision as seen from a moving helicopter. The student was asked to find the helicopter velocity which would make the final puck velocity be zero as seen from the helicopter. (This is the velocity of the center-of-mass of the two pucks.) The student picked a velocity for the helicopter, whose motion is shown by the small dot. When by trial and error the student has found the appropriate helicopter velocity there is a discussion of motion in the center-of-mass reference frame. This is a simple example of an open-ended simulation controlled by the student but imbedded in a tutorial context.

FIGURES

Fig. 1. Index page for a lesson on momentum conservation in collisions.

Fig. 2. An example of a tutorial question. The student may press the HELP key for the discussion shown in Figure 3.

Fig. 3. The beginning of a HELP sequence for the question shown in Figure 2.

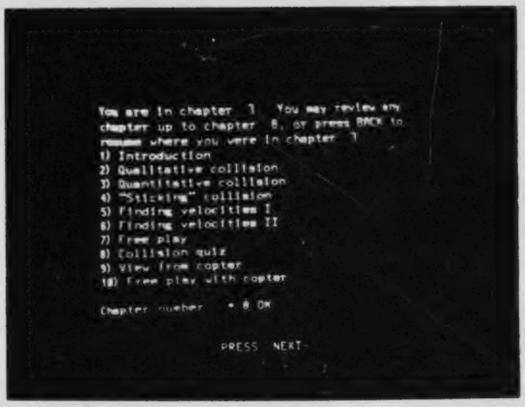
Fig. 4. A later step in the lesson. The student presses a LAB key to see the simulated collision shown in Figure 5.

Fig. 5. Simulated collision of two pucks.

Fig. 6. A quiz problem. The student may answer with any equivalent equation.

Fig. 7. Another quiz problem. The student answered incorrectly. He may finish the quiz or may go to the index page for review.

Fig. 8. A collision viewed from two different reference frames. The student chose a velocity for the helicopter, whose motion is shown by the small dot.

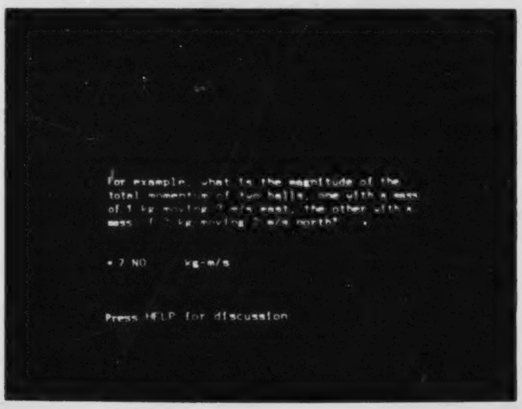


You are in chapter 1. You may review any
chapter up to chapter 8, or press BACK to
resume where you were in chapter 1.
1) Introduction
2) Qualitative collision
3) Quantitative collision
4) "Sticking" collision
5) Finding velocities I
6) Finding velocities II
7) Free play
8) Collision quiz
9) View from copter
10) Free play with copter

Chapter number = 8 OK

PRESS NEXT

Figure 1



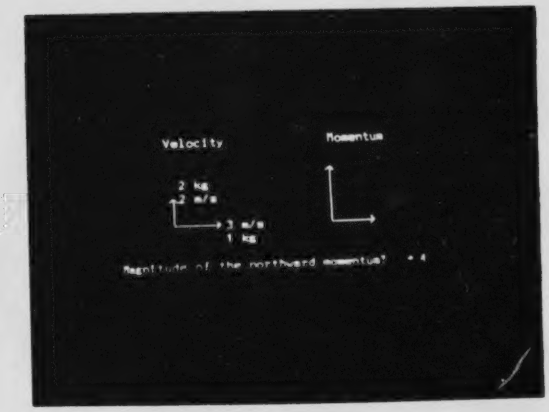
For example, what is the magnitude of the
total momentum of two balls, one with a mass
of 1 kg moving [illegible] m/s east, the other with a
mass of [illegible] kg moving 2 m/s north?

= 7 NO kg-m/s

Press HELP for discussion

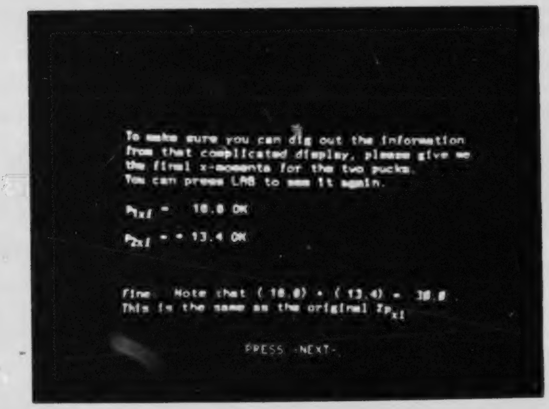
Figure 2

Figure 3



Magnitude of the northward momentum? 4

Figure 4



To make sure you can dig out the information
from that complicated display, please give me
the final x-momenta for the two pucks.
You can press LAB to see it again.

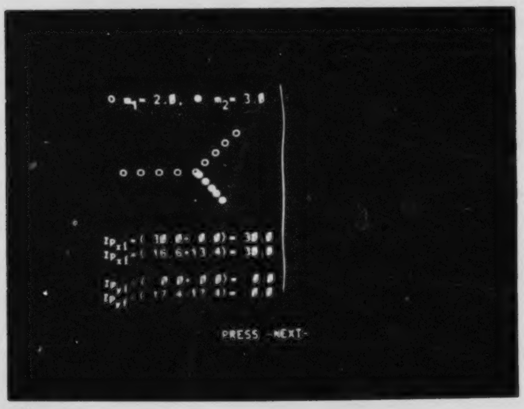
p_{1xf} = 16.6 OK

p_{2xf} = 13.4 OK

Fine. Note that (16.6) + (13.4) = 30.0
This is the same as the original Σp_{xi}

PRESS -NEXT-

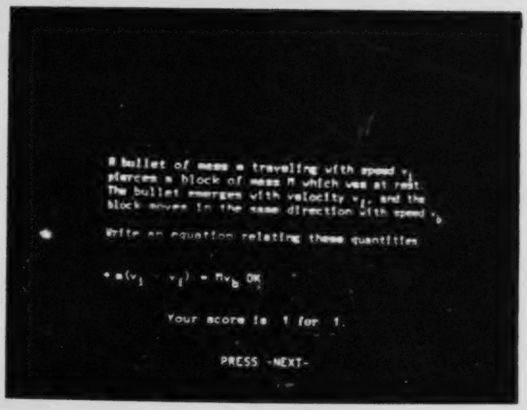
Figure 5



Σp_{xi}=(30.0+ 0.0)= 30.0
Σp_{xf}=(16.6+13.4)= 30.0

Σp_{yi}=(0.0+ 0.0)= 0.0
Σp_{yf}=(17.4-17.4)= 0.0

PRESS -NEXT-



A bullet of mass m traveling with speed v_i pierces a block of mass M which was at rest. The bullet emerges with velocity v_f, and the block moves in the same direction with speed v_b.

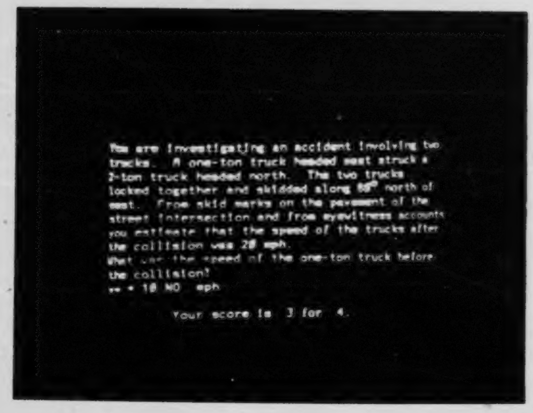
Write an equation relating these quantities

▸ $m(v_i - v_f) = Mv_b$ OK

Your score is 1 for 1.

PRESS -NEXT-

Figure 6



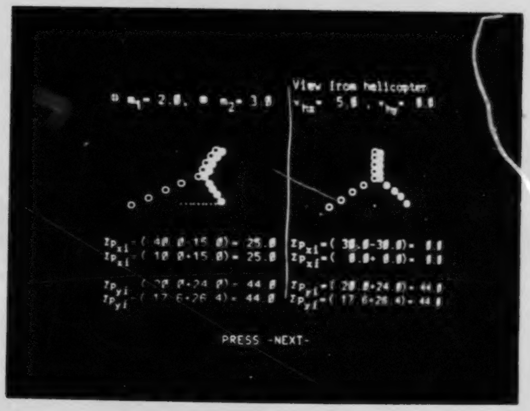
You are investigating an accident involving two trucks. A one-ton truck headed east struck a 2-ton truck headed north. The two trucks locked together and skidded along 60° north of east. From skid marks on the pavement of the street intersection and from eyewitness accounts you estimate that the speed of the trucks after the collision was 20 mph.

What was the speed of the one-ton truck before the collision?

▸ 10 NO mph

Your score is 3 for 4.

Figure 7



○ $m_1 = 2.0$, ● $m_2 = 3.0$

View from helicopter

$v_{hx} = 5.0$, $v_{hy} = 0.0$

$\Sigma p_{xi} = (40.0-15.0) = 25.0$

$\Sigma p_{xf} = (10.0+15.0) = 25.0$

$\Sigma p_{yi} = (20.0+24.0) = 44.0$

$\Sigma p_{yf} = (17.6+26.4) = 44.0$

$\Sigma p_{xi} = (30.0-30.0) = 0.0$

$\Sigma p_{xf} = (0.0+0.0) = 0.0$

$\Sigma p_{yi} = (20.0+24.0) = 44.0$

$\Sigma p_{yf} = (17.6+26.4) = 44.0$

PRESS -NEXT-

Figure 8

PHYSICS COMPUTER DEVELOPMENT PROJECT*
COMPUTER ASSISTANCE IN STUDENT PROBLEM ASSIGNMENTS

Mark Monroe
University of California
Irvine, California

This paper is a discussion of computer assistance in physics student problem assignments. Specifically, the development of dialog material for particular physics problems by the Physics Computer Development Project at the University of California is discussed.

THE PROBLEM ASSISTANCE IDEA

Computers have found heavy use in the physics curriculum at the University of California at Irvine. In addition to various computational uses, the development of extensive physics dialog material has been undertaken by the Physics Computer Development Project. The complete variety of this material is discussed in a separate paper (1). One aspect of this development problem assistance, is the subject of this paper.

The problem assignments for a physics course are indicative of the conceptual level of the course. The student work load at Irvine typically involves problem sets containing nontrivial exercises testing conceptual understanding, rather than simple formula plug-in assignments. Consequently, the student may start an assignment and run into difficulties on a particular problem. In such cases, the student may then reference related texts; or seek advice of colleagues or physics faculty. The problem may arise when such assistance is unavailable (e.g., late at night, on a weekend, etc.), and the student is then faced with the possibility of falling behind.

Computer assistance lends itself nicely to such situations. It is worth noting that computer assistance is also well suited to those students who will not seek faculty advice for fear of exposure. In a healthy terminal environment, the student has ready access to the interactive facilities of the computer. At Irvine, terminal usage is available essentially seven days of the week and late hours in the evening. In such an environment, computer problem assistance can be made available to the student when problems are still fresh.

The types of problem assistance are varied. Possibilities include complete dialog-type help, simple hints, and more complex assistance such as allowing the student to query the computer for reasonable approaches to a particular problem. The first two types of assistance have recently been employed at Irvine, while the latter, appearing most interesting, has yet to be realized. Another untried possibility is the medical-diagnostic approach, similar to one a doctor might take; this approach attempts to analyze student difficulties on the basis of his responses to diagnostically aimed questions. Completion of the analysis would be followed with review of and assistance with areas of difficulty.

The conversational or dialog problem help can itself take on a number of forms. These can range from an outline dialog with the student filling in most of the steps to more extensive programs in the vein of complete teaching conversations. The extent of the dialog, naturally, is correlated to the complexity of the problem. Proof problems, for example, may require extensive help, for generating proofs of physical principles is a nontrivial task. In problems where the student is given a starting point and some final result to verify, an outline or even a few hints may suffice. Hints or a short outline may be adequate for computational problems.

AN EXAMPLE OF COMPUTER PROBLEM ASSISTANCE--COMPTON

At Irvine, two problem dialogs have been developed for use with the freshman level relativity material. Based on problems from the Taylor-Wheeler Spacetime Physics, these dialogs deal with the problems of Compton scattering (COMPTON) and the Doppler shift (DOPPLER).

In the former, which will be discussed at some length, the student was given the initial relativistic expressions for the four-momentum of an incident photon and a stationary electron. He was then asked to derive the expression for the energy of the scattered photon as a function of the scattering angle. The problem tested the student's knowledge of the momentum-energy four-vector and the conservation laws of energy and momentum. Since these were relatively new concepts for many students, they represented areas of possible difficulty. This was the analysis used at the inception of COMPTON. The dialog was constructed with the idea of talking the student through the derivation, permitting him to invoke his judgments and choices at crucial points along the way (e.g., when to apply the conservation laws). The dialog was terminated with the student having obtained the energy and momentum equations necessary to complete the derivation. The algebraic details were left to the student.

The COMPTON dialog assumed knowledge of the problem and at least minimal understanding of the momentum-energy four-vector and the conservation laws. There was little remedial discussion of these concepts along the way. The structure of the approach involved permitting the student to progress through the derivation until he ran into difficulties. At trouble points, the problem was successively reduced to smaller subproblems of decreasing complexity, until a point was reached where the student could again proceed essentially on his own. Then, he was branched back up through the lattice, linking the subproblems together, until he reached his original trouble point. From there, the student continued with the derivation. Several attempts were permitted at most digressions, and hints were given before successive retries.

A simple illustration of the above technique is given in flowchart form in Figure 1. The portion of COMPTON illustrated there occurs early in the program. It is querying the student for the initial relativistic expression for the four-moment of the electron. The student is given half the expression and is prompted to complete it. After putting the student input into a standard formula form, a series of tests are performed to test the validity of the input. Units were chosen earlier so that the speed of light equaled one. This simplified subsequent work. Since this input occurred early in the program, it was felt that tests for c were warranted to get the student in the habit of using c = 1. Note that entering c does not affect the number of retries given to the student since it does not constitute an error input.

The input is then analyzed for the presence of a momentum term "p". Since the electron is initially at rest, this term is zero and should not appear. If a "p" is found in the input, a branch to a secondary loop is executed. Constituting a reduction to a subproblem for this simple case, this loop was included to lead the student into using the information that the electron is at rest, in obtaining his expression for its initial energy. The student is then asked to apply this piece of information in writing his energy expression.

If the correct response is found, it is acknowledged and the student continues. If not, and no other anticipated error inputs are found, the student is then given general hints. If he continues to fail to input a valid expression, he is given the correct expression and allowed to continue.

This simple example is representative of the philosophy used in COMPTON, the input analysis and branching routines were correspondingly more extensive for more complex inputs (e.g., the complete energy conservation equation for the system).

An example of how such a scheme worked in practice is given in Figure 2. This is the terminal output for the section of COMPTON flowcharted in Figure 1. This material is self-explanatory.

Handling of consecutive student inputs which cannot be adequately analyzed is an area warranting further discussion. The theory of the problem dialog is to help the student work the problem. Telling the student that he fails to have a feeling for the necessary principles involved has merit--if the area of difficulty can be sufficiently well defined. But, consecutive uninterpretable inputs really represent a more difficult task. Giving the student the correct response and permitting him to continue is one feasible scheme. Under such a scheme, subsequent student interaction may permit a more definitive analysis of his difficulties. Furthermore, there is pedagogical value in allowing the student to work the problem and analyze his own weaknesses upon completion.

A second philosophy involves pushing the student into consulting his instructor. It is not always necessary for the problem dialog to stand as a self-contained teaching conversation. Rather, providing a diagnosis of student difficulties may be sufficient. If these weaknesses are not well defined, referring the student to his instructor may be the only acceptable approach. The computer assistance may prove to have further value in student-instructor discussions. For example, the terminal printout from a problem assistance program may enable the instructor to better analyze student difficulties.

It is felt that the handling of uninterpretable responses in computer problem assistance requires an admixture of the aforementioned philosophies, with emphasis on student self-evaluation of weaknesses. Experience is the best guide in picking a legitimate scheme in such cases.

IRVINE USAGE OF COMPUTERS IN PROBLEM ASSISTANCE--COMPTON, DOPPLER, P5

COMPTON provided the first computer problem assistance for the Irvine physics students. Employed in conjunction with the 1969-70 freshman level course, the dialog received moderate usage. Student opinion appeared to be mixed, with the favorable response in the slight majority. This evaluation is based on the only evidence available--student course evaluations--and is to be taken with some reservations.

As with all initial run dialogs, COMPTON suffered from various technical difficulties, which included inadequate student input analysis and branching mechanisms. The major content criticism received was a failure to complete the problem. That is, it was found that some students were unable to complete the derivation with the energy and momentum equations obtained in the dialog. Further assistance with the final bits of algebra was requested. This was received as a legitimate criticism, for the algebra is a little messy for a freshman level course.

Other students found COMPTON tedious. These usually proved to be the better students for whom the dialog was not originally intended. These students usually obtained the important results before taking the dialog. This criticism was not totally without merit. Efforts were made to increase the interaction at the time of revision.

Another interesting, yet different, response came from the student who solved the problem, but not to his own complete satisfaction. This student seemed to view the COMPTON dialog as reinforcement. This was unanticipated, but has obvious pedagogical value.

In discussions between the author and Dr. Alfred Bork, the course instructor and chief project investigator, following COMPTON's first student usage, various revisions were outlined. Direct student feedback to Dr. Bork revealed another area of unexpected difficulty. Many students failed to have a firm idea of what the problem entailed. As a result of these discussions, extensive revisions in COMPTON were initiated. These included more extensive and elaborate analysis of student input; and inclusion of two entirely new sections: 1) an introduction querying the student on the specifics of the problem, and 2) an optional closing section discussing the algebraic details necessary to complete the derivation. The revised version of COMPTON, COMPTON2, has been readied for use with the 1970-71 course.

The second problem dialog currently available to the Irvine student is the DOPPLER dialog. A product of student feedback during the 1969-70 course, DOPPLER was initiated by Dr. Bork, and was explicitly formulated during the discussion between Dr. Bork and the author at the termination of the spring quarter. It is expected that DOPPLER will receive more usage in its first run (1970-71) than the original COMPTON. The feeling is, that being the result of student feedback, DOPPLER is more tailored to student needs. In general, for problems assigned in the normal fashion, it is felt that formulation of problem assistance dialogs on the basis of student-faculty interaction is preferable to those based on anticipation of areas of conceptual difficulty.

A second form of problem assistance presently employed at Irvine is the use of a physics newsfile (P5). The assistance consists mainly of hints for the weekly problem assignments. Co-authored by Alfred Bork and Chris Hill, a physics undergraduate having completed the course, the newsfile is updated late in the week after most students have already at least tried the problems. A variety of styles are employed (hints, outlines of reasonable problem strategies, etc.) in an effort to obtain a feeling for what the students find most helpful. Since the problem sets often involve use of the computer in a computational mode, the student may access the P5 newsfile while engaged in his problem set. It is felt that this type of assistance suffices for some problems, while supplying a starting point for more challenging problems, despite failing to make extensive use of the interactive possibilities.

AREAS FOR FURTHER DEVELOPMENT

Perhaps the ideal form of problem assistance is one making heavy use of the interactive facilities of the computer. Dialogs are definitely interactive, yet at the same time retain a linearity that restricts the student interaction. Elaborate uses of internal branching can greatly reduce these restrictions, but other forms of interactive material are seen. Two possible forms are discussed below.

One form involves the computer querying the student about his difficulty with a particular problem in much the same way an instructor might query a student in his office. This would include questioning the student about what he thinks the problem entails; what physical principles are relevant; what he has done with the problem so far; and what he thinks his difficulty might be. In turn, the student could query the computer on various aspects of the problem and what might constitute a reasonable problem strategy. This type of assistance requires complex mechanisms for analysis of student input and appropriate computer response.

A preferable, yet no less complex, scheme would allow the student to start off by querying the computer on approaches to his problem. This could even entail permitting the student to formulate his approach at the terminal. The student could pick a strategy and tell the computer what operations to perform. Simultaneously, he would give the computer the facts and concepts he felt pertinent. The computer would then give the student the results. By inclusion of provisions for keeping him within the general ballpark, wide latitude could be given the student--for example, restrictions on the operations and information which are acceptable. In addition, analysis of the strategies used might facilitate diagnosis of student difficulties. These could then be discussed at the terminal, still leaving the interaction heavily with the student. The complexity of such a scheme raises questions about its desirability. The tendency for the student to develop a shotgun tactic rather than quantitative methods under such a system also raises a pedagogical question about its merit.

In any computer problem assistance scheme, emphasis should be placed on development of logical methodology, developing the heuristics of problem solving. There is some merit to leading the student with a scheme that gets him to attack his problems as a set of subproblems. In addition, such approaches help to clearly define student weaknesses; revealing areas for further study or drill. After solving the subproblems, the formulation of a complete solution from the partial solutions develops relations between the basic principles of physics. This type of strategy is applicable to all problems in physics, and is essential to development of sophisticated physical intuition. Hence it is felt that any scheme of problem assistance that emphasizes these principles, and permits heavy student interaction constitutes a viable philosophy.

REFERENCES

*This project was supported by the National Science Foundation

1. Bork, Alfred M., "Physics Computer Development Project--A Progress Report".

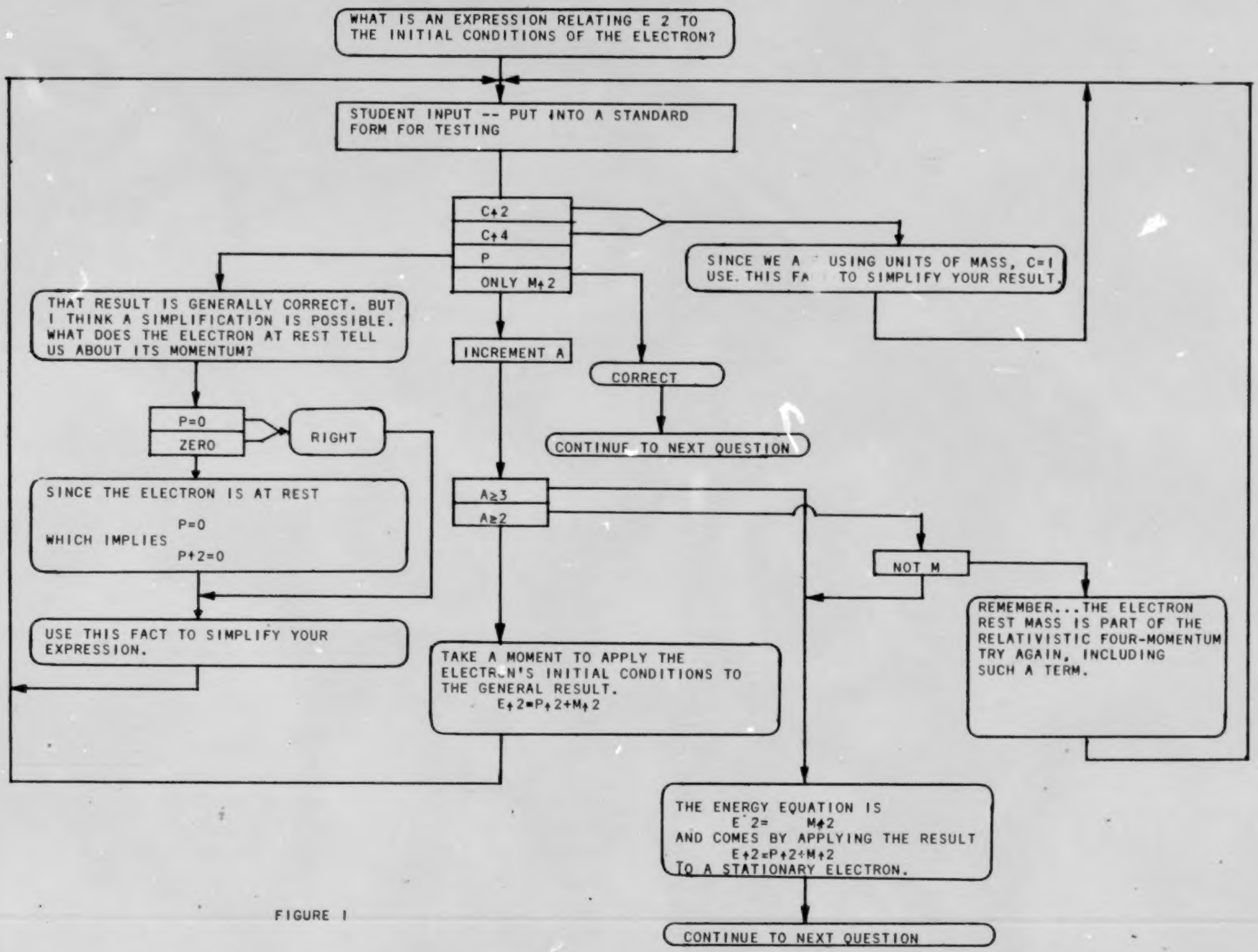

WHAT IS AN EXPRESSION RELATING E 2 TO THE INITIAL CONDITIONS OF THE ELECTRON?
STUDENT INPUT -- PUT INTO A STANDARD FORM FOR TESTING
C↑2
C↑4
P
ONLY M↑2
SINCE WE A USING UNITS OF MASS, C=1 USE. THIS FA TO SIMPLIFY YOUR RESULT.
THAT RESULT IS GENERALLY CORRECT. BUT I THINK A SIMPLIFICATION IS POSSIBLE. WHAT DOES THE ELECTRON AT REST TELL US ABOUT ITS MOMENTUM?
INCREMENT A
CORRECT
CONTINUE TO NEXT QUESTION
P=0
ZERO
RIGHT
SINCE THE ELECTRON IS AT REST P=0 WHICH IMPLIES P↑2=0
A≥3
A≥2
NOT M
USE THIS FACT TO SIMPLIFY YOUR EXPRESSION.
TAKE A MOMENT TO APPLY THE ELECTRON'S INITIAL CONDITIONS TO THE GENERAL RESULT. E↑2=P↑2+M↑2
REMEMBER...THE ELECTRON REST MASS IS PART OF THE RELATIVISTIC FOUR-MOMENTUM TRY AGAIN, INCLUDING SUCH A TERM.
THE ENERGY EQUATION IS E 2= M↑2 AND COMES BY APPLYING THE RESULT E↑2=P↑2+M↑2 TO A STATIONARY ELECTRON.
CONTINUE TO NEXT QUESTION

FIGURE 1

```
WHAT IS AN EXPRESSION RELATING E↑2 TO THE
INITIAL CONDITIONS OF THE ELECTRON?

E↑2 = P↑2*C↑2+M↑2*C↑4                       (STUDENT INPUT)

SINCE WE ARE USING UNITS OF MASS, C=1.
USE THIS FACT TO SIMPLIFY YOUR RESULT.

E↑2 = P↑2+M↑2                               (STUDENT INPUT)

THAT RESULT IS GENERALLY CORRECT, BUT
I THINK A SIMPLIFICATION IS POSSIBLE.
WHAT DOES THE FACT THAT THE ELECTRON IS
AT REST TELL US ABOUT ITS MOMENTUM?

? P=0                                       (STUDENT INPUT)

RIGHT.
USE THIS FACT TO SIMPLIFY YOUR EXPRESSION.

E↑2 = M↑2                                   (STUDENT INPUT)

CORRECT.                                    (A GOOD RESPONSE IS
NOW GIVE ME A SIMILAR EXPRESSION FOR THE     ACKNOWLEDGED, AND THE
INITIAL PHOTON ENERGY.                       STUDENT CONTINUES)

EPH↑2 = PPH↑2                               (STUDENT INPUT)

FINE.
SINCE THE PHOTON IS A ZERO REST MASS
PARTICLE, THERE IS NO MASS TERM.

NOW THAT YOU HAVE EXPRESSIONS FOR THE INITIAL
ELECTRON AND PHOTON ENERGIES WOULD YOU LIKE
SOME HELP WITH THE CORRESPONDING FINAL ENERGIES,
OR SHALL WE CONTINUE WITH THE CONSERVATION LAWS?

?CONSERVATION LAWS                          (STUDENT INPUT)
```

FIGURE [illegible]

STUDY OF WAVES USING IBM 1130 COMPUTER

Paul A. Smith
Coe College
Cedar Rapids, Iowa

This paper concerns itself with a number of computer programs written to support instructional activities in the physics laboratory relating to wave phenomena of a variety of sorts. Most of the programs were written for a group of exceptional high school students attending an NSF Summer Institute in which the Physics Program focused on wave phenomena. While the programs are all in the computational mode an effort has been made to structure activities to stimulate student initiative and participation even though on-line interaction is not present.

Following the very brief descriptions of the individual programs given below you will find a discussion of some of the instructional considerations which come to mind on thinking back over the local experience in using these programs.

INTFR is designed to assist students in internalizing the basic concepts of wave interference via assistance in using phasors graphically in a study of the expected interference pattern of up to thirty point sources of wave located along the y-axis. The point sources may be positioned to simulate a single or low order multiple slit or some other interesting pattern. A "screen" of specified extent may be located parallel to the y-axis at a specified distance from it along the positive x-axis. The printout is a record of input information, screen positions for 100 points uniformly distributed over the specified interval, and the phase angles of waves reaching each screen position from each point source specified. Overlaid on this is a graphical plot of the corresponding intensity pattern.

FERMT is a program designed to help students understand Fermat's Principle through an educated search for the point of cross-over of a beam of light or any other object traveling between two points in two semi-infinite planes wherein speeds of propagation are different. The boundary between the two regions is defined to be the x-axis. The student defines the coordinates of a point below the x-axis and a point above the x-axis, the speeds of propagation below and above the x-axis, and an interval along the x-axis within which the cross-over point is assumed to exist. The printout is a record of the input data, some written instructions, and only ten values of transit time corresponding to ten uniformly spaced points along the x-axis within the specified interval. By successive trials the student is to locate the exact cross-over point.

SYNTH is a computer program to carry out Fourier synthesis of a function given a specification of spectral terms of sine and cosine form using either wave lengths or wave numbers and either actual individual term specification or an indication that a certain number of terms are to be distributed individually over a range of values of wave length or wave number with relative amplitudes specified by the curve of a tabulated function. The student may thus see the graphical printout of various combinations of sine and cosine functions and so get a feel for Fourier analysis by observing Fourier synthesis including the approximate transformation of a continuous function into its reciprocal space. The output consists of a copy of input data and the graphical plot on the line printer with an associated tabulation of the function generated.

WAVE is a program to carry out a 200-step numerical integration of the wave equation in one dimension for a particle bound in a potential well which is specified by a tabulated function. Input is a limited list of positions and corresponding potential energies sufficient in number to define the potential well to sufficient accuracy along with an indication of how many eigen-values and eigen-vectors are desired. Output is a plot of the potential well, the requested eigen-functions plotted, and the corresponding eigen-values. The line-printer-graphs include numerical tabulations of the plotted functions which may be used to test orthogonality. Students may include "perturbations" of a given well to study their effect on the wave functions generated. A "shelf" may be included in the well to note the effect when the energy rises just above the level of the shelf.

The following considerations seem important to the author as items to keep in mind while seeking to incorporate the computer into physics instruction: (1) An effort should be made to grant the student some genuine free choice in his efforts--even to the point of having him experimenting in an open-ended way asking his own questions and structuring his own experiments and then discovering that an existing computer program may be of help to him at some point.

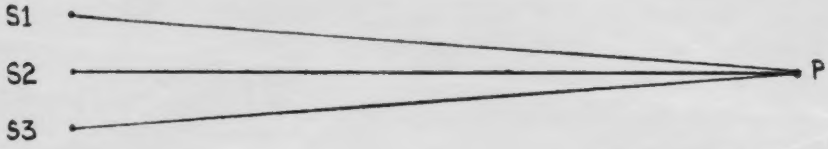
(2) In writing computer programs they should be designed so as to demand some thought and activity on the part of the student--leave some work for the student even though it might very easily be done by the computer in some cases. (3) The source deck listings of all programs should be available to the student with little effort required on his part to get them. In the case of massive programs such as the ones described here they should be available but not pressed upon the student; only the job they do needs to be explained. The author and a student assistant are developing a comprehensive system on the IBM 1130 to facilitate having even thousands of FORTRAN source decks available to students by name off a system with 8K Core and one disk drive--listings and/or punched decks. Interested individuals may write the author for further information. (4) The write-ups which follow are skeletal write-ups on the use of the programs and are adequate provided they are accompanied by personable explanations by an instructor to help novices get oriented. While much needs to be done to improve the quality of the supportive documentation one can engage in effective teaching with minimal documentation provided one is able to supply personal encouragement and support which will help the student gain a sense of being in charge of the situation rather than being a victim of the rigid demands of the computer. (5) The programs described here all use free-style input where possible to make the job for students in punching up data easier. Individuals not familiar with the author's system are referred to "Computer-Based Physics, An Anthology" which was published by the Commission on College Physics in 1969. (6) Even with all the efforts of computer enthusiasts there will be some students who will feel very cold towards the use of the computer; we must be sensitive to the emotional needs of those students. They should be helped to come to terms with what the computer is and what it is not in ways agreeable to themselves. (7) Massive computer programs such as those that the author has tended to write should be complemented by student use and modification of very simple ones in the style of Alfred Bork's "Computer Based Mechanics".

Source listings of the set of programs involved in the above set are available by writing to the author. The programs are in the FORTRAN language and written for use on an IBM 1130 with 8K core and 1/2 million 16-bit word single disk drive. Most of the programs involve a constellation of supportive subroutines. Owners of IBM 1130 systems may be interested in asking for compacted versions of the source decks which occupy about one tenth the number of cards as the actual source decks. The system for regenerating the source decks and/or listings will accompany the compacted decks. A contribution of $5.00 with each request will help cover costs of distribution.

The pages following contain actual documentation on the programs as it has been used with students. The author will appreciate receiving copies of any improved versions which any user may develop. Information about "bugs" in any of the programs will be appreciated, even though regretfully.

Phasor Addition

Suppose that there are three point sources of waves at locations "S1", "S2", and "S3". Suppose further that we are interested in the effect of these waves as they combine at a point "P". See the diagram below.



The amplitude of the waves at "P" will depend on both the passage of time and the different distances from the sources "S1", "S2", and" "S3" to "P".

Let us first consider the amplitude variation with time for waves from "S1" arriving at "P". Suppose that the distance from "S1" to "P" is an integer number of wave lengths and that at time t = 0 there is a crest at both "S1" and at "P". We may represent the amplitude at later times at "P" by the horizontal component of a line of unit length which is rotating about one end with a frequency equal to the frequency of the waves arriving at "P".

In a similar manner we may represent the wave amplitude due to waves originating at "S2" and "S3" by lines of unit length rotating about one end with the same rotational frequency.

There is no reason why these three lines should be parallel. In fact the angles between them should be determined by the relative amplitudes at "P" due to the separate waves arriving from "S1", "S2", and "S3" at some convenient time such as time zero. These relative amplitudes will depend on the different distances which the different waves have to travel to reach the point "P".

In our special case where the distance from "S1" to "P" is an integer number of wave lengths the angles between the lines or phasors for "S2" and "S1" on the one hand and "S3" and "S1" on the other hand will correspond to 360 degrees times the fraction of a wave length beyond an integer number of wave lengths that the corresponding travel distances are found to be.

In summary: "Phasors" are lines of unit length making angles relative to the horizontal which are equal to 360 degrees times the fractional part of a wave length beyond an integer number that the distance from corresponding wave source to location on a screen is.

The instantaneous amplitude on the screen due to a set of waves can be obtained by stringing the corresponding phasors tail to head with appropriate angles maintained. The horizontal component of the net geometrical distance from the tail of the first phasor to the head of the last will be the instantaneous amplitude. The actual distance between said initial tail and said final head will be the maximum possible instantaneous amplitude and the square of it will correspond to the intensity or brightness of the wave disturbance at the given point "P". Brightness and intensity always correspond to the rate at which energy is brought to the point in question by the wave disturbance, be it water, sound, light, radio, or microwave radiation that is under consideration.

The computer program "INTFR" gives a printout which can be the basis of an instructive exercise in adding phasors with the printout giving you an immediate check on your understanding. The location of the "*" gives you the expected relative brightness in terms of the maximum possible.

Test your understanding of phasor addition by checking 20 successive lines of output from the program INTFR in a region of some interesting brightness feature.

Purpose: To simulate the interference of light wavelets from a set of holes (located along the y-axis) given the interference takes place at a set of points (located evenly spaced along a line parallel to the y-axis at x = D) wherein the distance D measured in units of the wave length is not in excess of about 10,000.

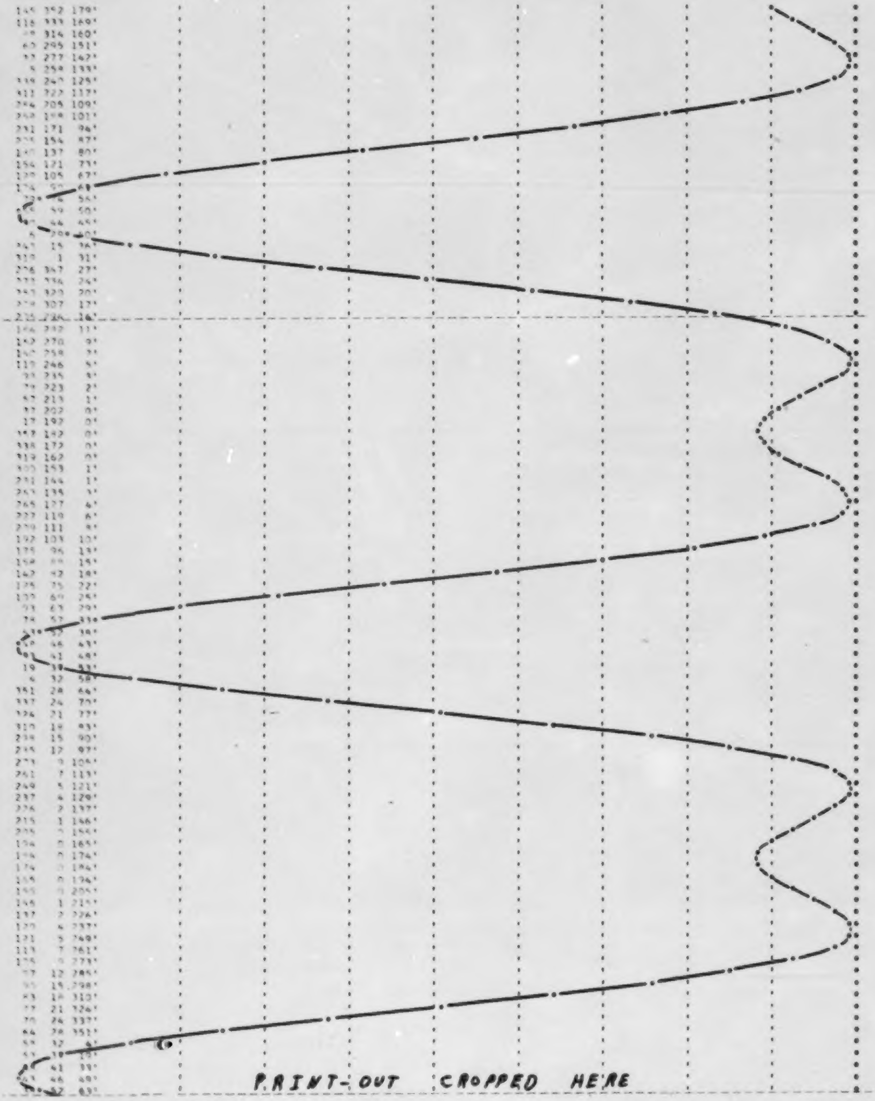
Method: Numerically carry out operations corresponding to the graphical addition of phasor vectors of equal magnitude with phase angles corresponding to the fractional part of a wave length in the path length from hole where the wavelet originates to the location on the screen where it and other wavelets interfere.

Use:
```
// JOB T
// XEQ INTFR
A card with any useful identification information on it must be here.
Card with wave length punched in columns 1-10 with explicit decimal point.
Card with two-digit-integer punched in columns 4 and 5 for number of holes.
Card with hole positions punched in 8F10.5 format; columns 1-10, 11-20, etc. up to 30
     values and so up to three cards for hole locations.
Card with the distance D in columns 11-20 with explicit decimal point and on same card in
     columns 1-10 the distance above and below the x-axis that the "screen" is to extend
     with explicit decimal point punched.
// * JOB END
```

Output: Copy of input data with locations on screen given in detail with your useful identification at the top of the page. Next, a continuous graph-tabulation listing the phasor angles in degrees for each screen position, one screen position per line of printed output, with phasor angles running from hole one on the left as the paper comes off the printer. The brightness for each screen position is indicated by a "*" on its appropriate line with zero at the right edge of the paper.

Switches: Computer Center Personnel may set switch zero up to cause a PAUSE during which printer carriage control may be disengaged.

```
PHASOR ADDITION EXERCISE

DISTANCE FROM HOLES PLATE TO SCREEN =    100.00000
WAVELENGTH OF RADIATION =         1.000000000

HOLES POSITIONS
  10.000000    0.000000  -10.000000

SCREEN POSITIONS
 -20.000000  -19.724137  -19.448275  -19.172413  -18.896551  -18.620689  -18.344827  -18.068965  -17.793103  -17.517241
 -17.241379  -16.965517  -16.689655  -16.413793  -16.137931  -15.862068  -15.586206  -15.310344  -15.034482  -14.758620
 -14.482759  -14.206896  -13.931034  -13.655172  -13.379310  -13.103448  -12.827586  -12.551724  -12.275862  -12.000000
 -11.724137  -11.448275  -11.172413  -10.896551  -10.620689  -10.344827  -10.068965   -9.793103   -9.517241   -9.241379
  -8.965517   -8.689655   -8.413793   -8.137931   -7.862068   -7.586206   -7.310344   -7.034482   -6.758620   -6.482758
  -6.206896   -5.931034   -5.655172   -5.379310   -5.103448   -4.827586   -4.551724   -4.275862   -4.000000   -3.724137
  -3.448275   -3.172413   -2.896551   -2.620689   -2.344827   -2.068965   -1.793103   -1.517241   -1.241379   -0.965517
  -0.689655   -0.413793   -0.137931    0.137931    0.413793    0.689655    0.965517    1.241379    1.517241    1.793103
   2.068965    2.344827    2.620689    2.896551    3.172413    3.448275    3.724137    4.000000    4.275862    4.551724
   4.827586    5.103448    5.379310    5.655172    5.931034    6.206896    6.482758    6.758620    7.034482    7.310344
   7.586206    7.862068    8.137931    8.413793    8.689655    8.965517    9.241379    9.517241    9.793103   10.068965
  10.344827   10.620689   10.896551   11.172413   11.448275   11.724137   12.000000   12.275862   12.551724   12.827586
  13.103448   13.379310   13.655172   13.931034   14.206896   14.482758   14.758620   15.034482   15.310344   15.586206
  15.862068   16.137931   16.413793   16.689655   16.965517   17.241379   17.517241   17.793103   18.068965   18.344827
  18.620689   18.896551   19.172413   19.448275   19.724137   20.000000

PHASES IN DEGREES AND PRINT OUT IN STARS OF INTENSITY
```



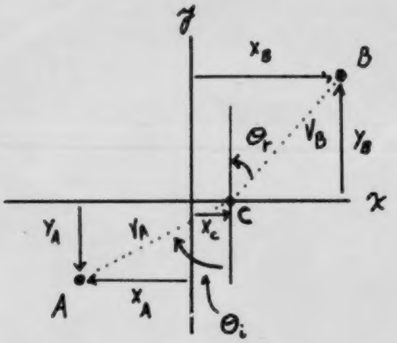
PRINT-OUT CROPPED HERE

The Principle of the Quickest Path

One of the most powerful summary statements about how light behaves is: "Light which travels from point "A" to point "B" follows a path which causes it to take either a minimum or maximum length of time compared to other paths which are almost like the actual path. Usually the actual path requires a minimum amount of time." This is "Fermat's Law" after Pierre de Fermat whose life overlapped those of Harvey, Descartes, Galileo, Kepler, Snell, Torricelli, Huygens, and Pascal in the early 1600's. Isaac Newton claimed that Fermat helped to inspire in him the development of the calculus.

This project will make it possible for you to gain an understanding of the power of Fermat's principle with the aid of modern computational aids and some simple experimental equipment.

The geometry of the situation we will consider is indicated below. We will consider how long it would take light to get from point "A" to point "B" if it were to travel in a straight line from "A" to a point on the "x" axis. The cartesian coordinates of "A", "B", and "C" are indicated.



The values are positive if to the right or up, negative if to the left or down. The distance from "A" to "C" is:

$$\sqrt{(X_C - X_A)^2 + Y_A^2}$$

and the distance from "C" to "B" is:

$$\sqrt{(X_B - X_C)^2 + Y_B^2}$$

The time to travel from "A" to "C" is:

$$\sqrt{(X_C - X_A)^2 + Y_A^2} \div V_A$$

and the time to travel from "C" to "B" is:

$$\sqrt{(X_B - X_C)^2 + Y_B^2} \div V_B$$

Thus the time to travel from "A" to "B" while taking straight line paths from "A" to "C" and from "C" to "B" is therefore:

$$\frac{\sqrt{(X_C - X_A)^2 + Y_A^2}}{V_A} + \frac{\sqrt{(X_B - X_C)^2 + Y_B^2}}{V_B}$$

We seek that value of X_C which will make this time as small as possible. (Why not seek a maximum?) We will solve our problem by computing the time for a variety of values for X_C.

At this point students who are not sharp on their algebra and arithmetic may feel like giving up! There is no need to give up as today we have programmable computers to help us. In this case there is a program available on the IBM 1130 called "FERMT" which will help compute the time to travel from "A" to "B" if the numerical values for X_A, Y_A, X_B, Y_B, V_A, and V_B are first fed in and then a list of 10 values of X which you wish to have used. Here is a sample of how a deck of computer cards should be set up for a run on the computer.

```
Card  123456789      These indicate column numbers on the computer cards.
  1   // JOB
  2   // XEQ FERMT
  3   C1 PUNCH YOUR NAME ON THIS CARD WITH "C" IN COLUMN ONE AND "1" IN COLUMN TWO
  4    XA=-10, YA=-20, XB=50, YB=30, VA=1.0, VB=1.5  (Column 1 blank)
  5    -20 -10 0 10 20 30 40 50 60 70 = TRIAL VALUES OF XC.  (Column 1 blank)
  6   // * JOB END
```

Note carefully: Card 1 and card 2 must be exactly as indicated. Start in column one and include blank spaces exactly as indicated. Card 3 must have a "C" in column one and a "1" in column two. Card 4 must have column one blank. The non-numeric characters are optional but the values must be in the order indicated. Card 5 must have ten values on it. Decimal points may be used. Card 6 must be exactly as indicated, including blanks.

When you have your deck prepared have a fellow student check you out on it and follow any oral instructions you have been given about getting it run on the computer. Use the results of the computer computations to make improved estimates of the values of XC which might yield the shortest travel time--repeating the process until you have XC to better than 0.1% accuracy. (Question: How will you be able to tell if you have the needed accuracy? Talk it over with your fellow students to work out your own method of being sure you have the desired accuracy.)

You are free to pick your own values for the coordinates of "A" and "B" and the speed of the light in region "A" and in region "B". It is suggested that you make all your values in the range from 0.0001 to 10000.0 for simplicity in keypunching, however. (Question: What should you do about "units" that are usually associated with distances and speeds? Again talk it over with your fellow students, remembering that the computer is doing only arithmetic for you.)

Suggestion: For fun you might try to compute "by hand" or "by slide rule" the time to take one of the paths calculated by the computer--to check up on the accuracy of the computer!

Suggestion: You may find it very interesting to become familiar with the EPIC 3000 electronic desk computer in the Statistics Laboratory and use it to carry out the above travel time computations. There are manuals on the desk with the EPIC 3000 telling you how to program the computer. Look especially for the sheet headed "PRINCIPLES OF OPERATION OF THE EPIC". and the sheet headed "PRINCIPLES OF AUTOMATIC OPERATION OF THE EPIC". At the end of these notes you will find a suggested program for computing the travel times of this project. Can you develop a slightly different program to accomplish the same job? Work one out with your fellow students that may also be interested in such a project.

Suggestion: Compare the results of computations on the IBM 1130 with those on the EPIC 3000. Are they identical? If they are not identical, why not? Talk it over with fellow students and when you have agreed on a reason check with your instructor to see if you are right.

Suggestion: The speed of light in a vacuum is the "speed limit" for light. It travels slower in any transparent material. Make up a reasonable vacuum "speed limit" for your problem and calculate the ratio of the speed limit to the speed of region "A" and the speed of region "B". These are the "indices of refraction" appropriate to regions "A" and "B". Now test Snell's Law which says that the product of the index of refraction in region "A" and the sine of the angle of incidence is equal to the product of the index of refraction in region "B" and the sine of the angle of refraction. Notice that these two angles are measured from a line perpendicular to the boundary between the regions at the place the ray of light crosses over. To get these ideas clear in your mind talk them over with other students.

Experimental: You will be provided with some slabs of glass, pins, and flat boards into which you can stick pins to sight along. By placing two pins on one side of a slab of glass and two pins properly oriented on the other side you can view all four pins as if they were in a straight line by looking through the slab of glass edgewise. By talking with other students in the class work out your own methods of testing the predictions made by the use of Fermat's Principle and Snell's Law. The following questions may help you organize your thinking:

1. How can you use pins to define the direction of a ray of light?
2. How can you simplify the problem to that of light being bent on passing through only one glass-air boundary?
3. How can you get the ratio of the speed of light in air to the speed of light in the glass?
4. What significance does the ratio of the two indices of refraction have?
5. What should be the value of the ratio between the sines of the angles of incidence and refraction?
6. Does Fermat's Principle approach the problem with the image of a ray of light entering the glass at a certain point and traveling in a certain direction?
7. Does Snell's Law approach the problem from the point of view of light that starts at one point and eventually gets to another previously selected point beyond the boundary at which the speed of the light changes?

Report: Become involved in your own way in computational and experimental activities dealing with Fermat's Principle, Computational Aids, and Snell's Law. Spend three to five hours on this activity and write up a brief (no more than one page) report indicating what you did and what you learned. Hand the report in.

FERMT

```
C1  TEST OF FERMT PROGRAM

XA =    -10.00000   YA =    -20.00000

XB =     50.00000   YB =     30.00000

VA =      1.00000   VB =      1.50000

THE ABOVE ARE YOUR INPUT VALUES.  BELOW ARE THE TRAVEL
TIMES WHICH CORRESPOND TO YOUR CROSS OVER POINTS.
PLOT TIME AGAINST CROSS OVER POINT TO GET AN IMPROVED SET
OF POSSIBLE CROSS OVER POINTS.  THEN RE-RUN PROGRAM WITH
THIS NEW SET OF POSSIBLE CROSS OVER POINTS.
GET CROSS OVER POINT TO 0.1 PERCENT ACCURACY AND THEN
TEST SNELLS LAW FOR BEST CROSS OVER POINT.

        XC            T
   -20.00000     73.13250
   -10.00000     64.72135
     0.00000     61.23369
    10.00000     61.61760
    20.00000     64.33979
    30.00000     68.75837
    40.00000     74.93350
    50.00000     83.24555
    60.00000     93.88295
    70.00001    106.49913
```

Note: A program FERMS is also available which graphs out "T" versus "XC".

SAMPLE OF EPIC PROGRAM FOR FERMAT'S LAW

For use on an electronic programmable desk top calculator, in this case the Monroe EPIC 3000.

Programming and first computation

```
START RESET                     ■
PRESS LEARN
XA =              10.0000
ENT
XC =              50.0000  -
-
RPT                             -

X                               ×

YA =              20.0000-
RPT
X                               ×

+                               +

√                               r

PRINT             44.7213  ◊ DA

VA =               1.2000  ÷
÷
PRINT             37.2677  ◊ TA

STORE                           S

CLEAR                           C

XC =              50.0000
ENT
XB =              80.0000  -
-
RPT                             -

X                               ×

YB =              30.0000-
RPT
X                               ×

+                               +

√                               r

PRINT             42.4264  ◊ DB

VB =               1.5000  ÷
÷
PRINT             28.2842  ◊ TB

REC C                           R

+                               +

PRINT             65.5519  ◊ T

PRESS AUTO
```

While in Automatic Mode

```
XA =              10.0000
ENT
XC =              50.0000
ENT
YA =              20.0000
ENT
                  44.7213  ◊ DA

VA =               1.2000

                  37.2677  ◊ TA

XC =              50.0000
ENT
XB =              80.0000
ENT
YB =              30.0000
ENT
                  42.4264  ◊ DB

VB =               1.5000
ENT
                  28.2842  ◊ TB

                  65.5519  ◊ T
```

STUDY OF WAVES

Purpose: To synthesize a function from a user's specification of a set of spectral components in terms of either their wave lengths or in terms of their wave numbers; in either case amplitudes of sine and/or cosine terms may be specified explicitly or by way of a few representative amplitudes.

Intent: To graphically illustrate the relationship between a Fourier transform and the function it represents as in the case of wave functions in position space and momentum space and the associated topics of wave packets and Heisenburg uncertainty.

Use:

```
// JOB T
// XEQ SYNTH
```

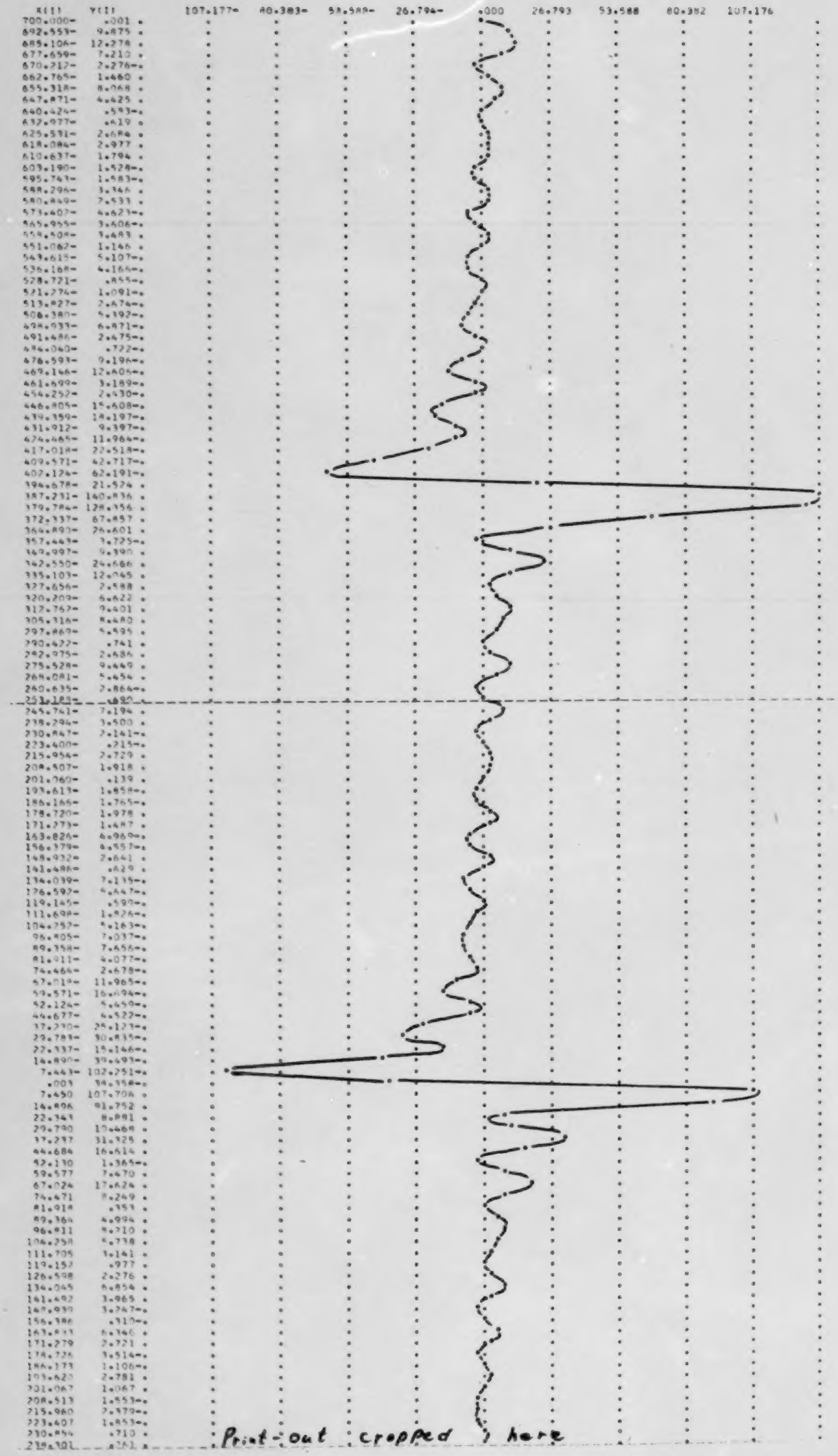
1. C1 Any identification information you wish starting with "C" in column one and "1" in column two.
2. Free style listing of wave lengths (or wave numbers) for which spectral terms in sine and/or cosine will be requested; end with 9999 after last entry for entries to be used as wave lengths; end with 0.0 9999 after last entry for entries to be used as wave numbers. Use as many cards as necessary but make no more than 50 entries before the terminal sign "9999".
3. Free style listing of coefficients of sine functions of wave lengths (or wave numbers) specified by entries preceding; the association being made between the N-th entries in th' two lists. Make as many entries in this list as in the last and end with 9999.
4. Free style listing of coefficients of cosine functions as in item No. 3 just preceding. Proofread these listings carefully!
5. Free style entries of the following quantities in order:
    - Lower limit on X for graphed function,
    - Upper limit on X for graphed function,
    - Number of spectral terms.
        - If number of spectral terms specified here is identical to the number of entries under 2, 3, and 4; then exactly those entries will be used.
        - If number of spectral terms specified here is different from the number of entries under 2, 3, and 4; then those entries will be treated as representative points on functions specifying the sine and cosine amplitudes and actual amplitudes will be obtained by interpolating the functions at equal intervals between the extreme values listed in entries at #2.

```
// * JOB END
```

Output: Record of input data and a three page tabulation and graphical representation of the function specified by user in above input data. In tabulation algebraic minus signs follow integer digits on the right side of numbers rather than to the left as customary.

Note: In Items 2, 3, 4 entries in corresponding ordinal positions will be associated with each other regardless of the pattern of numerical values.

```
C1 TEST OF SYNTH WITH SQUARE BUMP IN WAVENUMBER SPACE

TABULATION OF WAVELENGTHS AND COEFFICIENTS FOR COSINE AND SINE TERMS FOR THESE WAVELENGTHS

INTERPOLATION USED IF NECESSARY TO GET THE REQUESTED     40  SPECTRAL TERMS

WAVELENGTH    COSINE      SINE
 NUMBER


   0.00000   0.00000   1.00000
   0.02564   0.00000   1.00000
   0.05128   0.00000   1.00000
   0.07692   0.00000   1.00000
   0.10256   0.00000   1.00000
   0.12820   0.00000   1.00000
   0.15384   0.00000   1.00000
   0.17948   0.00000   1.00000
   0.20512   0.00000   1.00000
   0.23076   0.00000   1.00000
   0.25641   0.00000   1.00000
   0.28205   0.00000   1.00000
   0.30769   0.00000   1.00000
   0.33333   0.00000   1.00000
   0.35897   0.00000   1.00000
   0.38461   0.00000   1.00000
   0.41025   0.00000   1.00000
   0.43589   0.00000   1.00000
   0.46153   0.00000   1.00000
   0.48717   0.00000   1.00000
   0.51282   0.00000   1.00000
   0.53846   0.00000   1.00000
   0.56410   0.00000   1.00000
   0.58974   0.00000   1.00000
   0.61538   0.00000   1.00000
   0.64102   0.00000   1.00000
   0.66666   0.00000   1.00000
   0.69230   0.00000   1.00000
   0.71794   0.00000   1.00000
   0.74358   0.00000   1.00000
   0.76923   0.00000   1.00000
   0.79487   0.00000   1.00000
   0.82051   0.00000   1.00000
   0.84615   0.00000   1.00000
   0.87179   0.00000   1.00000
   0.89743   0.00000   1.00000
   0.92307   0.00000   1.00000
   0.94871   0.00000   1.00000
   0.97435   0.00000   1.00000
   1.00000   0.00000   1.00000

GRAPH WILL BE PRINTED OUT FOR  X  VALUES FROM  -70.00001  TO   70.00001
```


X(I)
Y(I)
107.177-
80.383-
53.588-
26.794-
.000
26.793
53.588
80.382
107.176
Print-out cropped here

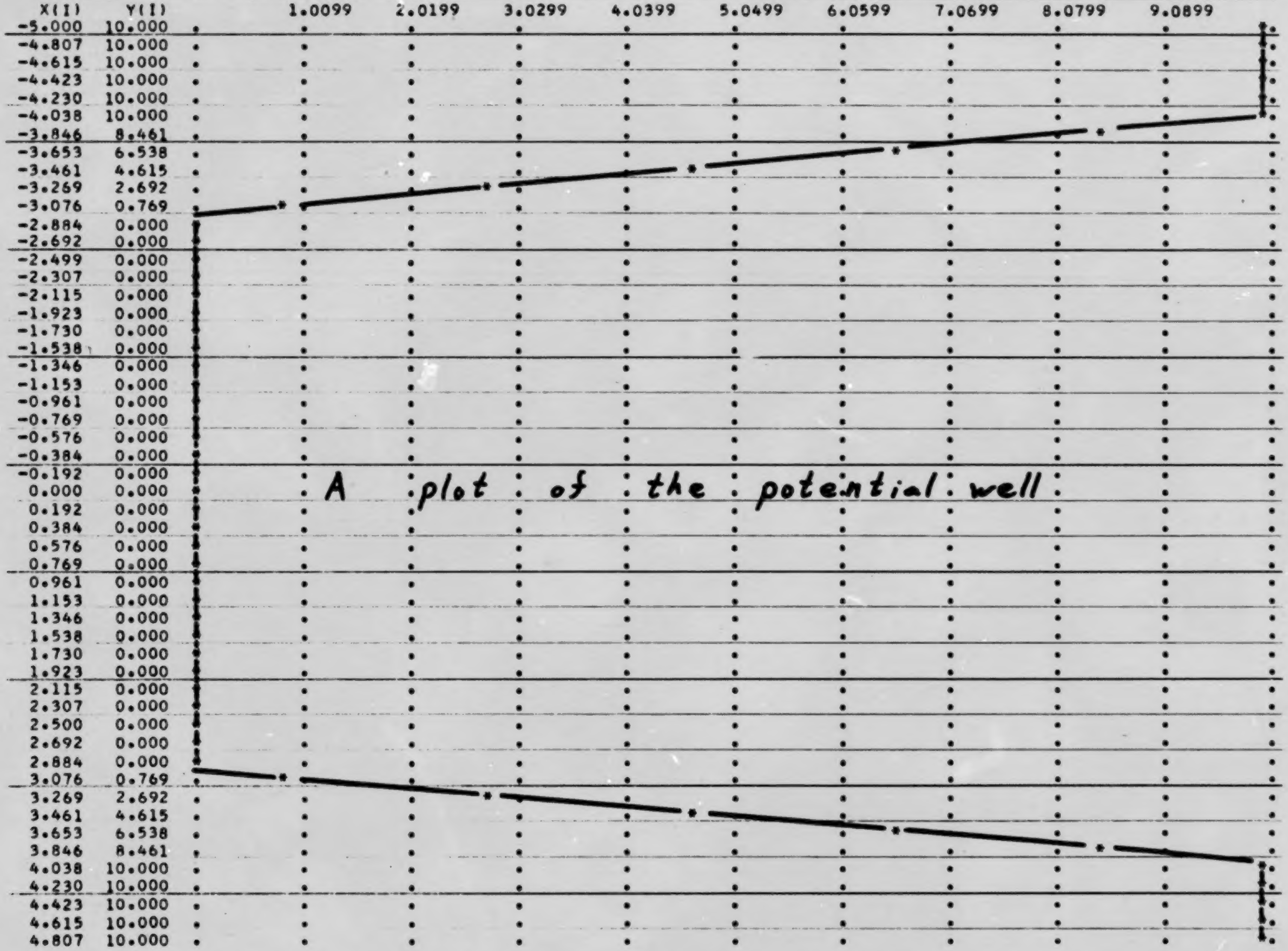
Purpose: To give the user the eigen-functions (in graphical and tabulated form) and the corresponding eigen-values for the low energy states of a quantal particle in a specified one-dimensional potential well.

Use:
```
// JOB
// XEQ WAVE
C1  (Punch your name and any desired comments here)
  Punch the number of eigen functions you desire on this card.
  Free-style punch X(1) V(2) X(2) V(2) ... X(n) V(N)
  On a final card punch four nines in a row 9999
Additional sets of data cards just as above may be added on.
// * JOB END
```

Suggestion: A well which is 10 units wide and 20 units deep and having the "hole" restricted to the middle third of the range with steep walls will give satisfactory results. The well sides must be thick enough to include a decay to essentially zero of the eigen-function. Try different shape wells, wells of nearly the same shape, wells with a shelf on the side, wells which are exactly symmetric, wells which are far from symmetric.

Note: The potential well must be defined by not more than 25 pairs of values. You may run into trouble if you ask for more eigen-functions than there are bound states!

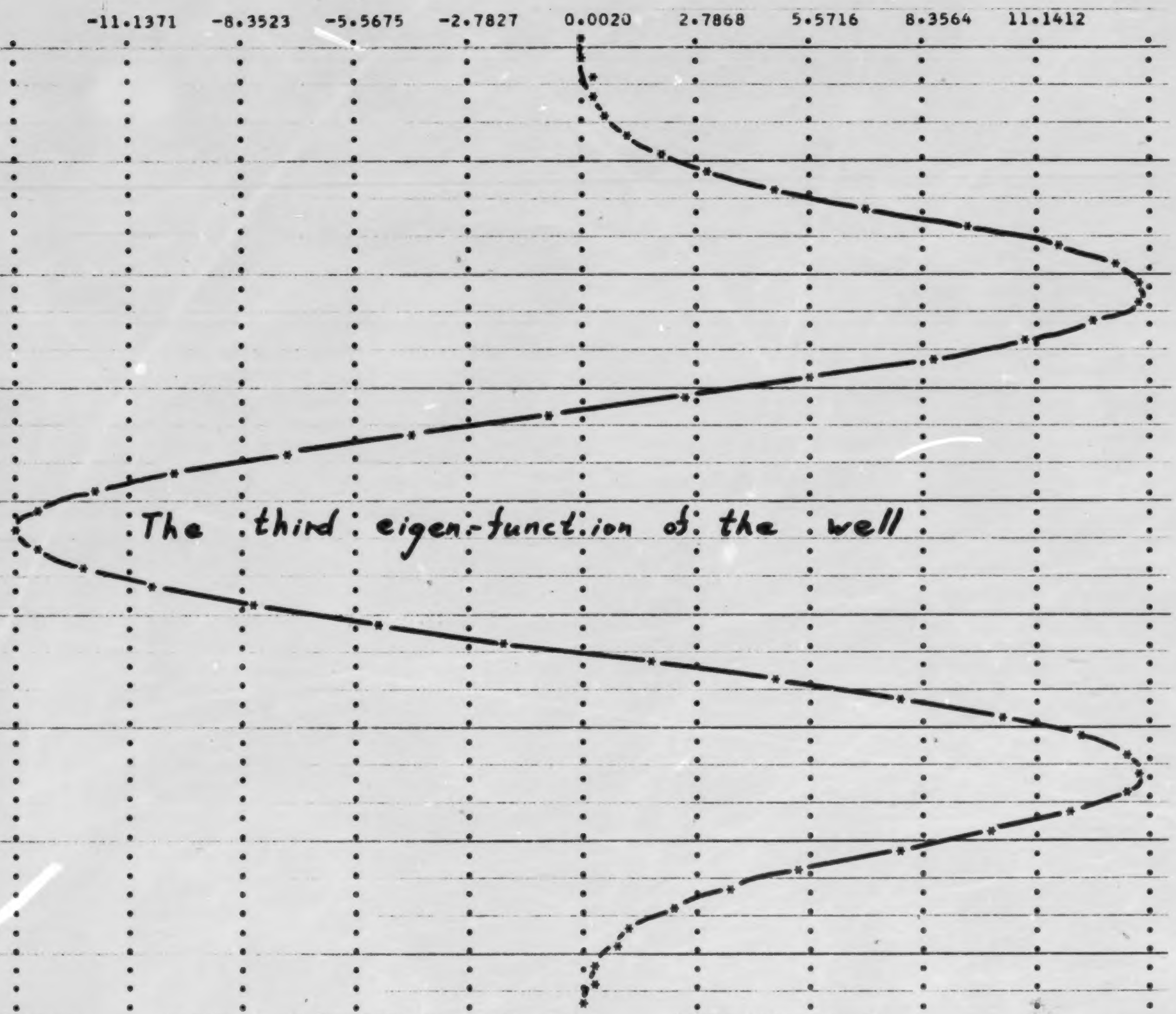
Suggestion: Test eigenfunctions corresponding to different energies to see if they are orthogonal. Test predictions of perturbation theory on eigen-functions of two wells of very nearly the same shape.



```
 X(I)    Y(I)          1.0099  2.0199  3.0299  4.0399  5.0499  6.0599  7.0699  8.0799  9.0899
-5.000  10.000
-4.807  10.000
-4.615  10.000
-4.423  10.000
-4.230  10.000
-4.038  10.000
-3.846   8.461
-3.653   6.538
-3.461   4.615
-3.269   2.692
-3.076   0.769
-2.884   0.000
-2.692   0.000
-2.499   0.000
-2.307   0.000
-2.115   0.000
-1.923   0.000
-1.730   0.000
-1.538   0.000
-1.346   0.000
-1.153   0.000
-0.961   0.000
-0.769   0.000
-0.576   0.000
-0.384   0.000
-0.192   0.000
 0.000   0.000
 0.192   0.000
 0.384   0.000
 0.576   0.000
 0.769   0.000
 0.961   0.000
 1.153   0.000
 1.346   0.000
 1.538   0.000
 1.730   0.000
 1.923   0.000
 2.115   0.000
 2.307   0.000
 2.500   0.000
 2.692   0.000
 2.884   0.000
 3.076   0.769
 3.269   2.692
 3.461   4.615
 3.653   6.538
 3.846   8.461
 4.038  10.000
 4.230  10.000
 4.423  10.000
 4.615  10.000
 4.807  10.000
```

A plot of the potential well

```
  X(I)      Y(I)
-4.925     0.000
-4.734     0.079
-4.544     0.184
-4.353     0.346
-4.163     0.615
-3.973     1.073
-3.782     1.854
-3.592     3.082
-3.401     4.832
-3.211     7.062
-3.021     9.513
-2.830    11.645
-2.640    13.085
-2.449    13.745
-2.259    13.586
-2.069    12.619
-1.878    10.895
-1.688     8.517
-1.498     5.634
-1.307     2.415
-1.117    -0.948
-0.926    -4.257
-0.736    -7.308
-0.546    -9.924
-0.355   -11.948
-0.165   -13.260
 0.025   -13.784
 0.215   -13.475
 0.405   -12.365
 0.596   -10.518
 0.786    -8.044
 0.976    -5.090
 1.167    -1.829
 1.357     1.538
 1.548     4.814
 1.738     7.804
 1.928    10.329
 2.119    12.235
 2.309    13.409
 2.500    13.783
 2.690    13.336
 2.880    12.095
 3.071    10.129
 3.261     7.700
 3.451     5.376
 3.642     3.489
 3.832     2.128
 4.023     1.239
 4.213     0.710
 4.403     0.400
 4.594     0.213
 4.784     0.092
```



The third eigen-function of the well

```
ENERGY =      1.65502667       MULTIPLY WAVE FUNCTION BY    0.1E 04          EXITED STATE IS  2
```

INTRODUCTION AND APPLICATION OF COMPUTER PROGRAMMING IN COMMUNITY COLLEGE SCIENCE AND TECHNOLOGY PROGRAMS

P. Hlavaty
Orange County Community College
Middletown, New York

Our Science and Technology faculty decided in 1968 that a concentrated effort would have to be made to introduce the Digital Computer into our Engineering, Physics, Mathematics, and Technology curricula. We discovered that our graduates were finding a need for these skills upon transferring to four-year colleges or upon their entrance into industrial positions as technicians. Our first attempt to provide such service was to request our own Computing Center for assistance, only to find that it was primarily involved with administrative functions and student records, and not equipped to handle some one hundred and fifty students with elementary FORTRAN programs requiring extensive debugging.

Our next approach was to turn to a Time-Sharing arrangement with a private company via teletypes and phone lines. This was placed in operation during 1968 and we taught a Digital Computations course for the first time that summer, utilizing two teletypes with paper tape transmission of the students' programs, in order to provide the most economical service. The results were that our costs went up and student interest went down! It cost us approximately $7,000 for telephone time, teletype rental and computer time--3 to 4 hours per week total, for all students. This was clearly unsatisfactory. The students lost interest in typing a program on paper tape, they did not attempt to introduce it into the classroom (nor did the faculty who were unwilling or unable to take the time to learn how to use the computer or assign problems in this area) and our budget went into the red.

During the Spring of 1970, we discovered that Federal funds might be available for the purchase of a minicomputer (Data General Nova) and also that a Time Sharing arrangement was available through Dartmouth College--the costs for rental of a port on their computer could be reduced by sharing the time and telephone line with several other colleges or secondary schools in the area. The Federal grant was approved and during the Fall of 1970, our own computer became operable and available to all students at any time of the day.

This particular minicomputer enabled us to operate four teletypes simultaneously, utilizing the BASIC language. The larger Dartmouth computer was available some twenty hours each week for the use of our engineering students writing FORTRAN programs. This was a far cry from the expensive Time Sharing arrangement with a private corporation.

We found that we could introduce all Science and Technology students to BASIC programming immediately during the Fall semester and discovered that these skills made the teaching of FORTRAN much less difficult. Student interest blossomed and the availability of a computer giving instantaneous response at all hours of the day, maintained this interest. The minicomputer system, complete with four teletypes and 8K memory, costs $16,000 and the Dartmouth rental is costing approximately $225 per month for some 80 hours of computer time. Our budget has been halved and our computer time increased by a factor of five not including the unlimited time on the minicomputer! Both systems have functioned without major problems for some five months now. An additional bonus was realized when we discovered that the paper tape programs from the NOVA minicomputer were compatible with the Dartmouth computer. Engineering students could prepare their FORTRAN programs on tape off line and run as many programs as they desired during the hours the Dartmouth computer was available. Each student has his own user number and storage space on the Dartmouth system.

The constant availability of the minicomputer has encouraged students and faculty to utilize this device in other courses. We find that students have been developing programs to be used in their engineering, chemistry, mathematics, and physics courses. Physics students now find that programs can be written in BASIC which speed the tedious calculations sometimes needed for data reduction in their physics laboratory courses. A Physics instructor has programmed the minicomputer to guide the student through the necessary calculations involving vector analysis in force or velocity vector laboratory problems. These programs can be stored and accessed on command by the student.

Mathematics and Physics students can graph most functions simply by utilizing a "TAB" command in BASIC which moves the teletype printer to any position (horizontally) calculated by the function. Examples of such programs may be seen on the following pages in addition to a photograph of the system.

Our technology students spend two semesters learning many different aspects of programming including the BASIC and FORTRAN languages, plotting routines, string variables, accessing library programs available from the Dartmouth computer, and practicing the solution of Mathematics and Physics problems via the minicomputer. Projects are also utilized in an effort to encourage the students to program more difficult problems by working in small groups and dividing responsibility for major portions of the program.

We have discovered that the system stimulates our Math/Science students so that more than a few are now contemplating Computer Science as a career. Our better students are encouraged to continue by taking additional courses as electives and given the freedom to explore the Dartmouth computer library or propose projects of interest to them. Special courses are to be offered for the entire faculty introducing the system to them in such a way as to encourage use of the system, if only to access the Dartmouth library and gain information in the Social Sciences, Business, Mathematics, Engineering or Statistics.

It should be remembered that all of this has come about during a period of five months!

The minicomputer system has the capability to grow along with the student population since additional memory (up to 32K) and more teletypes (up to 16) can be added as needed. We are expanding our course sections next year to accommodate all students from any department who wish to take a course in computer programming. It will now become, we hope, a popular elective,and introduce the computer to the entire student population as well as the entire faculty.

ADDENDUM

It should be noted that the NOVA minicomputer can be programmed to create numerical control tapes and our mechanical technology students will soon be introduced to this subject. It is also possible to provide remote access on campus and future plans include the installation of teletypes in Chemistry, Technology and Biology Departments.

NOTE

The programs on the following pages were submitted to Professor D. Davies in his Digital Computations I class, Fall, 1970 at Orange County Community College. They represent projects completed by freshmen science and technology students as part of the normal programming course.

PURPOSE:

THIS PROGRAM WILL INTEGRATE A FUNCTION OF X WITH THE NORMAL RESTRAINTS PLACED ON DEFINITE INTEGRALS

INSTRUCTIONS:

TO USE THIS PROGRAM TYPE:

```
00100  let Y = (YOUR FUNCTION OF X)
```

USE 'BASIC' EXPRESSIONS.

SAMPLE PROBLEM:

```
00100  LET Y = X↑2 + 1
RUN
APPROXIMATE ANSWER FOR INTEGRAL
LOWER LIMIT, UPPER LIMIT
? 1,2
APPROXIMATE INTEGRAL EQUALS 3.585
program by TOM ARMSTRONG
STOP AT 0210
```

PURPOSE:

THIS PROGRAM CHANGES ANY BASE 10 NUMBER TO THE CORRESPONDING NUMBER IN THE BASE OF YOUR CHOICE.

INSTRUCTIONS:

TO USE THIS PROGRAM TYPE RUN. THIS STARTS THE PROGRAM. WHEN ASKED FOR THE NUMBER AND BASE, TYPE IN THE NUMBER YOU WANT CHANGED AND THE BASE TO WHICH YOU WANT IT CHANGED.

THE ONLY RESTRICTIONS ARE THAT BOTH THE NUMBER AND BASE MUST BE POSITIVE INTEGERS LESS THAN 10,000,000. WHEN YOU ARE DONE WITH THE PROGRAM PRESS THE ESC KEY ON THE TELETYPE.

SAMPLE PROGRAM:

```
RUN
WHAT IS THE NUMBER THAT IS TO BE CHANGED? 345
WHAT BASE IS THE NUMBER TO BE CHANGED TO? 2
1  0  1  0  1  1  0  0  1
CARE TO TRY AGAIN?
WHAT IS THE NUMBER THAT IS TO BE CHANGED? 23456
WHAT BASE IS THE NUMBER TO BE CHANGED TO? 12
1  1  6  10  8
CARE TO TRY AGAIN? ESC
STOP AT 0040
```

(PROGRAM BY W. SHAVE)

PEAK VOLTAGE CALCULATION

PURPOSE:

THIS PROGRAM WILL CALCULATE PEAK VOLTAGE WITH INSTANTANEOUS VOLTAGE, FREQUENCY, RESISTANCE AND TIME GIVEN.

INSTRUCTIONS:

TYPE RUN
INSERT VALUE FOR INSTANTANEOUS VOLTAGE
    "   "   " FREQUENCY
    "   "   " TIME
PEAK VOLTAGE VALUE IS PRINTED.

SAMPLE PROBLEM:

```
RUN
PEAK VOLTAGE CALCULATION

INSTANTANEOUS VOLTAGE (E) = ? 5
FREQUENCY (F) = ? 800
TIME (T) = ? .0005
PEAK VOLTAGE = 8.50656
PROGRAM BY ROMAN RODRIGUEZ
STOP AT 0260
```

PSEUDO-RANDOM WALKER

PURPOSE: THIS PROGRAM REVEALS THE POSITION THAT A MAN FOLLOWING A RANDOM COURSE WITH VARYING DISTANCES WILL FIND HIMSELF.

INSTRUCTIONS: TO USE THIS PROGRAM TYPE: RUN. ALSO SUPPLY VALUES FOR L AND H. IF YOU WISH FURTHER RESULTS TYPE IN "YES" IN RESPONSE TO THE QUESTION. THEN TYPE IN NEW VALUES FOR L AND H. IF SATISFIED WITH THE RESULTS TYPE "NO".

SAMPLE PROBLEM: FIND THE FINISHING POINT WITH VALUES OF 50, 50 FOR L AND H.

SAMPLE SOLUTION: ALL THE INFORMATION THAT THE USER MUST SUPPLY IS UNDERLINED.

```
RUN

              P S E U D O - R A N D O M   W A L K E R

(LAKE)........................ SCHOOL ...................(HOME)
       -L-STEPS                        H-STEPS>

HOW LARGE DO YOU WANT L,H?
?50,50

RESULT
SAFE HOME


DO YOU WANT MORE VALUES FOR L,H?
YES OR NO?
? YES
HOW LARGE DO YOU WANT L,H?
? 47,48


RESULT
SPLASH.  INTO LAKE.


DO YOU WANT MORE VALUES FOR L,H?
YES OR NO?
? YES
HOW LARGE DO YOU WANT L,H?
? 30,25


RESULT
SPLASH.  INTO LAKE.


DO YOU WANT MORE VALUES FOR L,H?
YES OR NO?
? NO

STOP AT 0290
```

(PROGRAM BY M. BRODERICK)

UNIVERSAL GRAVITATIONAL CONSTANT

PURPOSE

THIS PROGRAM CALCULATES THE UNIVERSAL GRAVITATIONAL CONSTANT. THE DATA USED IN THIS PROGRAM IS FROM THE CAVENDISH EXPERIMENT. THE EQUATION USED IN THIS PROGRAM IS $G = M^2 x D x S x B^2/(T^2 x L x M)$.

INSTRUCTIONS

TO USE THIS PROGRAM TYPE RUN. THIS COMMAND STARTS THE PROGRAM. AFTER COMPUTER PRINTS THE PROBLEM STATEMENT, IT WILL ASK FOR THE FIRST DATA ITEM. AFTER A QUESTION MARK, YOU MUST ENTER A NUMBER. AFTER TYPING THE FIRST DATA ITEM, PUSH THE RETURN BUTTON. THIS PROCEDURE IS CARRIED OUT FOR EACH DATA ITEM.

SAMPLE PROBLEM

CALCULATE A VALUE FOR THE UNIVERSAL GRAVITATIONAL CONSTANT, USING THE FOLLOWING DATA ITEMS:

B = .0476 D = .0494 M = 1.501 T = 686 S = 6.15 L - 154

SAMPLE SOLUTION

ALL DATA ITEMS TO BE SUPPLIED BY USER IS UNDERLINED. SEE SAMPLE OUTPUT FOR MEANING OF ABOVE STATEMENT.

SAMPLE OUTPUT

```
RUN

GRAV. ATTRACTION

   DISTANCE,B,BETWEEN BODIES? .0476
   DISTANCE,D, FROM CENTER OF CONNECTING BAR? .0494
   ATTRACTING MASS,M,IS?  1.501
   TIME,T, ELAPSED?  686
   DISTANCE,S,MOVED FROM INITIAL POINT?  6.15
   DISTANCE,L,FROM LIGHT SOURCE?  154
GRAVITATIONAL ATTRACTION = 6.24591E-11
SYSTEMATIC ERROR?  8.6
TRUE GRAVITATIONAL ATTRACTION=                6.78305E-11

PROG. BY M. MIGLIONICO

*READY
```

PAUL DANIELS

DIGT. COMP. PROJECT

PLOT RC CIRCUIT WAVESHAPING

PURPOSE THIS PROGRAM WILL ENABLE YOU TO PLOT THE CURVE FOR VOLTAGE ACROSS A CAPACITOR WHEN YOU SUPPLY DATA.

INSTRUCTIONS TO USE THIS PROGRAM THE USER SUPPLIES VALUES FOR RESISTANCE CAPACITANCE AND VOLTAGE.

TYPE ALL UNDERLINED INFORMATION AND USE BASIC EXPRESSIONS.

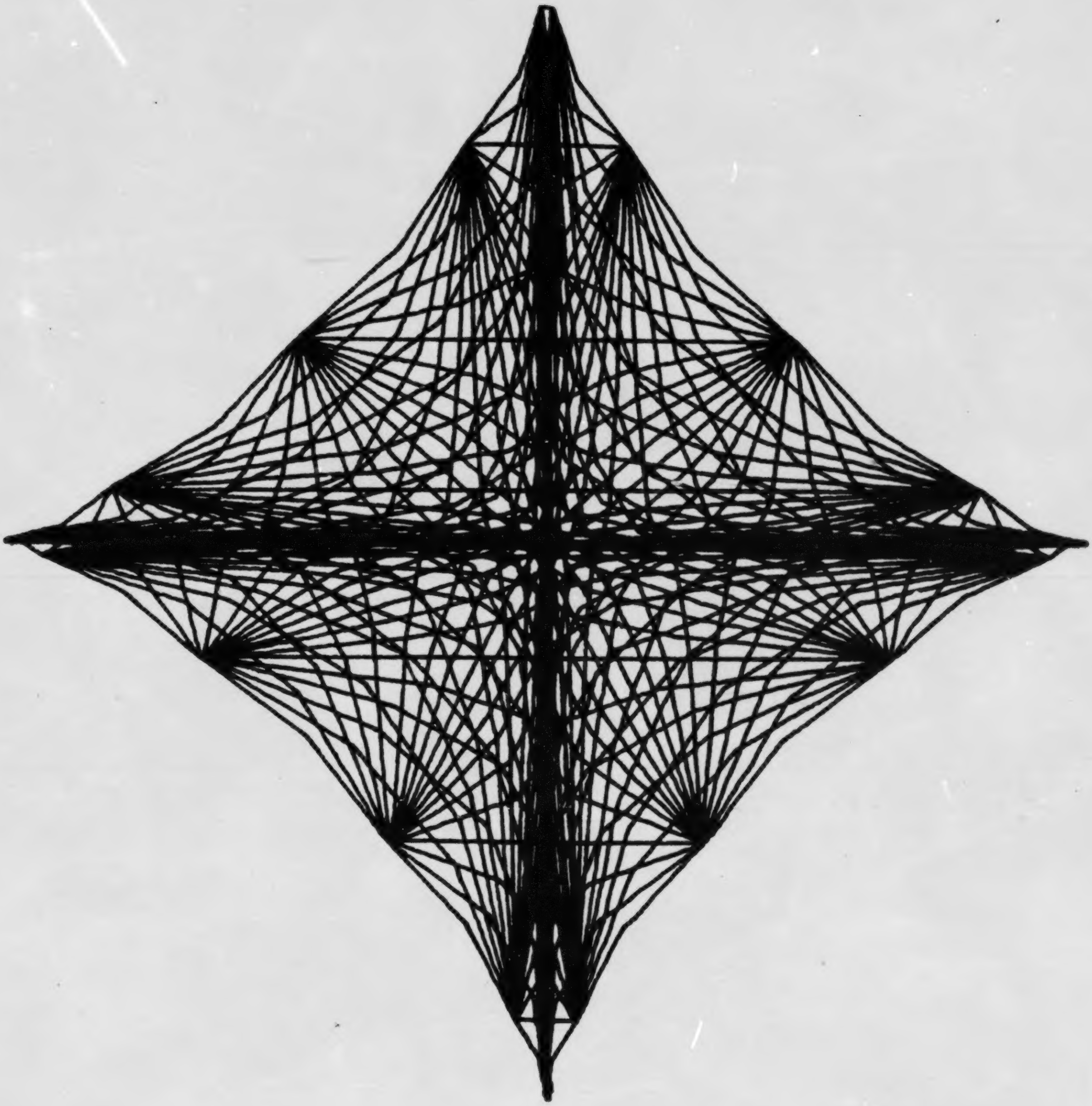
VC = ?

SAMPLE PROBLEM PLOT THE INSTANTANEOUS VOLTAGE ACROSS THE CAPACITOR FOR THE VALUES OF RESISTANCE, CAPACITANCE AND VOLTAGE GIVEN.

SAMPLE SOLUTION ALL INFORMATION TO BE SUPPLIED BY USER IS UNDERLINED. SEE SAMPLE OUTPUT.

SAMPLE OUTPUT

```
RUN

  P L O T   R C   C I R C U I T   W A V E S H A P I N G

THIS PROGRAM WILL ENABLE YOU TO PLOT THE CURVE
FOR CAPACITANCE WHEN YOU SUPPLY THE VALUES ASKED FOR


TYPE YOUR VALUES FOR TIME LOW, HIGH, INCREMENT? 6E-5,3E-3, 1E-4

TYPE YOUR VALUES FOR RES, CAP, VOITAGE?  1000, 1E-6,10

TIME         .........         VOLTAGE       .........
.00005          -                                                9.41764
.00015          -                         * * 8.52144
.00025          -                       *     7.71052
.00036          -                     *       6.97676
.00046          -                 * *         6.31284
.00056          -              **             5.71209
.00066          -           * *               5.16851
.00076          -         **                  4.67667
.00086          -       **      4.23162
9.59999E-4      -      *        3.82893
.00106          -      *        3.46456
.00116          -      *        3.13486
.00126          -      *        2.83654
.00136          -     *         2.56661
.00146          -    *          2.32237
.00156          -    *          2.10136
.00166          -    *          1.90139
.00176          -    *          1.72045
.00186          -   *           1.55673
.00196          -   *           1.40859
.00206          -   *           1.27454
.00216          -   *           1.15325
.00226          -   *           1.04351
.00236          -  *            .944205
.00246          -  *            .854352
.00256          -  *            .77305
.00256          -  *            .699484
.00276          - *             .63292
.00286          - *             .57269
.00296          - *             .518191
DO YOU WANT TO USE DIFFERENT VALUES, YES OR NO? NO
```



Albert B. Meador

THE TIME-SHARING COMPUTER IN THE BUSINESS SCHOOL CURRICULUM

J. Peter Williamson
Amos Tuck School of Business
Dartmouth College
Hanover, New Hampshire

Time-sharing came to the Amos Tuck School some five and a half years ago. In fact one might almost say that computing at the Amos Tuck School began five and a half years ago. For although the School possessed computing equipment before we turned to time-sharing, that equipment was seldom used beyond the usual routine course in FORTRAN programming, and it certainly made no impact on the curriculum.

In this paper I will describe some of what we have learned over these five and a half years, particularly in the field of finance and investments. I shall be discussing the experience of graduate MBA students, but whatever we have learned is equally applicable to the undergraduate business student. Few students arriving at the Tuck School have had any business education at the undergraduate level. Very few have had any experience with computers (in the years when we were just beginning to work with time-sharing probably none of the students had had any computer experience), and the entering Tuck student shares with undergraduate business majors a lack of quantitative skills and some resistance to acquiring them.

I'll try to organize my discussion around a number of questions:

1. What makes students use (or not use) computers?
2. Is the faculty or the student the key to expanded computer use?
3. What can computer availability do to an existing course?
4. What new courses may be called for?
5. Is there increased emphasis on quantitative and mathematical expertise?
6. What about "canned" programs?
7. Are we developing too much student reliance on the computer?

I will conclude with a specific example, an investments course which was developed to make maximum use of a time-sharing computer.

1. What Makes Students Use (or Not Use) Computers?

A few years ago, before Dartmouth College embarked on time-sharing, the Amos Tuck School possessed an IBM 1620, a very respectable batch processing computer. The machine, together with a card punch and an adequate supply of blank cards, was located in a room that was open to students 24 hours a day, seven days a week. Any student was free to help himself, the only formality being a request that he learn to operate the equipment under supervision before running it on his own. The room housing the equipment was conveniently located and for all practical purposes the equipment was the exclusive property of a little over 200 students. Almost none of them used it. That is, they did not use it beyond the single programming exercise required of all students in a FORTRAN programming course. A few students of course spent many hours with the machine. Perhaps half a dozen out of 200. The others simply could not see the point of mastering enough FORTRAN to be able to write useful programs without error, of punching cards, listing the cards on a printer to spot errors, punching corrected cards, listing the to compile and run the resulting program, receiving a series of error messages from the computer, and beginning the process all over again. The problem assignments in regular course work were always such that they could be handled without a computer, and only a true computer buff could be tempted to spend twice as much time solving a problem with a computer as was required using a desk calculator.

When the first time-sharing terminal was located in the Tuck School building the response was quite different. For the benefit of the reader who is not yet familiar with time-sharing I will venture on a very brief description.* Essentially, a single high speed central computer serves a number of people at terminals, usually standard teletype machines, accepting instructions from each in turn, performing the demanded calculations, and sending the results back to the appropriate terminal. All of this is done with enough dispatch that each terminal user is generally under the impression that he has exclusive use of the computer.

Perhaps the most obvious advantage of time-sharing is that it requires no punched cards. A program is written as fast as a programmer can type. Errors are corrected simply by retyping instructions. Once the program has been typed there is no carrying of cards to a computer and waiting for a turn. The programmer simply types "RUN," and the computer is in operation. If there are errors in the program the computer will respond with error messages and the programmer types his corrections.

Substitution of the time-sharing terminal for the traditional card punching process was an important element in changing student attitudes, but it was not the only one. The Dartmouth Time Sharing System from its inception made use of a new computer language, BASIC, in place of the traditional FORTRAN. Without going into details of programming languages I will simply note that whereas few students were competent to write even a simple FORTRAN program before the end of some weeks of instruction, the ordinary student can begin to write BASIC programs after two or three days of self study.

These two new features, the time-sharing terminal and the simple language, account for a complete reversal in the student attitude towards computing. In the Fall of 1965 our entering students were given the usual course in FORTRAN programming and were introduced to the IBM 1620, but they were also taught the rudiments of BASIC programming and were introduced to the time-sharing terminal. All of this took place within the confines of a course in computer programming. As yet, none of the other courses offered by the School had felt the impact of computing.

The brief introduction to time-sharing began to have an effect. It occurred to a number of first year students that an assignment calling for multiple regression might lend itself to a time-sharing program. The same assignment had been used year after year, but no student had ever attempted to handle it on the traditional batch processing computer. A hand calculated multiple regression is a very tedious operation. And the chance of avoiding error in the many calculations involved is very small. When it was discovered that the students who had resorted to the time-sharing terminal had not only cut short the drudgery of the assignment but had for the most part obtained correct answers, there was a new respect for time-sharing computing.

I think there are two lessons to be learned from this example, and from similar experiences that succeeded it. First, like the "critical mass" that must be present before nuclear fission will take place, there is probably a critical convenience level that must be present before students willingly take to computing. In shifting from what must have been one of the most convenient batch processing computer systems available for students, to a time-sharing system, we clearly passed that critical convenience level. Second, the most powerful inducement to students to make use of a computer is the discovery that it can relieve the drudgery and significantly improve the accuracy of regular work assignments. In this respect business school students are not very different from businessmen. My own experience is that one achieves little by introducing computers to a businessman as a device that makes new kinds of work possible. Show him that he can do what he is now doing faster and more easily, and you are likely to accomplish something. The new applications can come later.

2. Is the Faculty or the Student the Key to Expanded Computer Use?

Beyond the minimal formal instruction offered in BASIC programming and the provision of time-sharing terminals, the initiative in computer applications at the Tuck School came largely from the students and not from the faculty. The example I have described above was repeated over and over, as students discovered how easy it was to use the computer for assignments that faculty had been giving for years in the expectation that students would handle them using no more than a desk calculator. Indeed, in a few instances students who had developed some expertise in programming began to show faculty how much further some of the assignments could be carried once a computer was employed.

During the first year of time-sharing, some of the faculty, particularly those teaching Production and Finance, began to explore what this new facility could do for their courses. It was true then, just as it is generally true now, however, that students were not given assignments simply to force them to use a computer. I think this is an important point. It is easy enough to force a student to use a computer: simply give him an assignment and tell him it must be done on a computer. Most schools are familiar with this kind of assignment; it is part of any course in computer programming or computer application. Its impact is likely to be minimal. Students who completed their computer exercises in the days before the Tuck School had a time-sharing system were happy never to touch the computer again.

As time has gone on, the complexity of student assignments has tended to increase, to make full use of the computing facilities that are available. It is no longer quite fair to say that these assignments do not force a student to use a computer. The plain fact is that he could not handle them otherwise. It is still true however that assignments are not devised to occupy the student with the computer. They are devised to increase his understanding of the subject matter of a course. This is crucial, I believe, if the faculty is to be successful in encouraging students to fully exploit the opportunities in computing. But in the end the faculty can only present the opportunities; the student must take the initiative in exploiting them. And I sometimes think that with our more complex assignments we may have lost something of the self teaching that was so evident when the use of the computer was entirely optional and the student used it only because he appreciated what it could do for him.

3. What Can Computer Availability Do to an Existing Course?

I will now turn to my rather specific experience with the time-sharing computer in the teaching of Finance, and hope that those whose primary interest does not lie in Finance will bear with me. First, it occurred to me when I began to try out time-sharing that in the process of writing a rather simple computer program the student might better develop and display his understanding of an analytic process. In the course of teaching Finance to first year students I had always dealt with the calculation of bond prices and bond yields to maturity. The students made use of bond tables to answer fairly simple questions involving prices and yields, and hopefully they understood what lay behind the tables. As a practical matter it was simply not feasible to ask students to make the calculations without using the tables. During that first year of experience with the time-sharing system I required the first year students to write a computer program to calculate the price of a bond, given the yield to maturity, and to calculate the price correct to eight figures.

Now if you have always used tables to solve bond problems you probably have a fairly good idea of how the tables are put together. I thought so, but when I tried to write a computer program to give me answers correct to eight figures I discovered I knew less about the subject than I had realized. So did the students. To get an answer correct to eight figures you have to know exactly what you are doing. The approximations, the shortcuts, and the right general idea that were all sufficient when we never probed beyond the accuracy of a book of tables were no longer good enough. I think probably that class was the first of my Finance classes that really understood precisely how bond yields and prices are calculated. The computer programming skills required were negligible. A program of no more than 10 or 12 lines is quite sufficient to do the calculations. The difficulty arose entirely in the logic--understanding just how bond prices are deduced from bond yields and vice versa.

So much for a better understanding of what we had always been doing in the course. I discovered next that it was becoming feasible to give students far more realistic problems to solve than had been possible in the past. If a student is to be expected to handle assignments with hand calculations, even if desk calculators are available, a great many real life problems must be stripped down to bare and very much simplified essentials. Constructing cash budgets for a corporation, for example, usually involved presenting the students with a pared down set of financial accounts and a few very simple assumptions regarding what would probably happen to the corporation over the very near future. The result was an exercise which tested the ability of the student to manipulate accounts but which could never appear anything more than an exercise--it was impossible to see much practical value in preparing these budgets. Within two years of the introduction of time-sharing at the Tuck School all first year students, working in groups of five and six, were required to project, for an actual corporation, the annual income statements, balance sheets and cash flow statements for five years into the future. The projections were to be based upon past financial statements and were to reflect all the students could learn about their company that might have a bearing upon its future financial experience. The purpose was to spot future financial deficiencies, needs for short or long term financing, or potential financial crises. The assignment was infinitely more realistic than the earlier exercises had been. There was an obvious practical value to what the students were doing and indeed we began to invite corporate financial officers to attend student presentations and discuss with the students the results of their projections. We found enormously increased interest on the part of the students, a much greater understanding of the value of budgeting, and a much enhanced ability to do the budgeting.

Finally, I began to discover ways of carrying the discussion of familiar problems, and the understanding of analytic techniques, far beyond the levels that had been possible in the past. To take one example, I had for some years included in the Corporation Finance course a discussion of leasing as a financial device. The computations involved in analyzing a single lease problem, however, were so horrendous that I had been forced to deal with the problem by assigning it to a group of students who worked for many days with a desk calculator to prepare a set of calculations and presented them in class. We rarely got beyond a discussion of just how the calculations were done, there was little time to explore the implications of the calculated results and no possibility of exploring what the results might have been for a different set of assumptions. Once the time-sharing computer was available it became feasible to ask every student in the class to do in an evening's preparation what this group had in the past taken many days to do. It became reasonable to ask the students to interpret the results of their calculations and to repeat the calculations for a variety of assumptions. We were moving towards what I regard as one of the key aspects of almost any financial analysis--sensitivity analysis, which is the testing of the sensitivity of the results to the assumptions that were made. In the leasing case for example, where the ultimate question is whether a corporation should borrow money and purchase an asset or whether it should rent the asset, it becomes important to know how this decision may turn on the probable useful life of the asset, the interest rate that must be paid on a loan, the method of depreciation available for tax purposes and so on. Sensitivity analysis was almost out of the question before we had the time-sharing computer.

4. What New Courses May Be Called For?

The introduction of a substantial amount of computing may seem to call for a number of courses on computing and the computer. Indeed, some institutions appear to offer many courses about computers, but not much computing. I think it is important to decide whether students are to study computers and computing for their own sake, or whether the computer is simply to be regarded as a tool. I would guess that few business schools offer courses in library science, yet I suppose we all regard the library as an important resource for our students and as a potantial important resource for them after they graduate. At the Tuck School we have tended to regard the computer in the same way. Students are given a minimal introduction to the use of the computer (three class sessions are devoted to computer programming) and they are then left pretty much on their own. An elective second year course in computer systems is offered, but apart from this no effort is made to deal with computer science as such.

I offer no conclusion as to whether computer science belongs in a business school curriculum. My point is simply that the study of computer science is not a necessary consequence of the introduction of quite substantial computing power. In fact, because of the introduction of the BASIC computer language in place of FORTRAN, we now spend considerably less time teaching students how to use computers that we did in the old days of the largely unused batch processing equipment.

Our conscious avoidance of computer science has some drawbacks. By and large our graduates find it easy and natural to turn to a computer to solve problems, but they have very little feel for the economics of computing and almost no knowledge of the alternative computer configurations that may be available. The Dartmouth student, including the Tuck student, grows accustomed to computing power as a free commodity and is rarely required to consider the most efficient way of solving a problem, in terms of minimum computer cost. This can be a handicap when a student goes to work for a corporation that is far from oblivious to the cost of computing.

5. Is There Increased Emphasis on Quantitative and Mathematical Expertise?

It is important at this stage to distinguish between quantitative expertise and mathematical expertise. The Tuck School, by virtue of the introduction of the time-sharing computer, has become a rather quantitative school. What this means is that students do a great deal of figure manipulation. They may be performing a great many calculations on a few basic items of data. Or they may be processing a vast data bank, searching for relationships or verifying hypotheses. On the whole, however, the work at Tuck still does not involve much mathematical expertise.

Some mathematical techniques have been quickly transformed from curiosities into standard resources. The use of multiple regressions is an example. Regression analysis was taught at the Tuck School before we had the time-sharing computer. Each student was required to go through at least one regression analysis by hand. It seems fair to say that this was the last regression analysis most of those students ever expected to perform. One simply does not do multiple regressions by hand. He either does them on a computer or he does not do them at all. But with the advent of the easily accessible high speed computer, students tend to do a great deal of regression analysis. What was once a curiosity is now a standard tool. I do not believe that this has increased the need for students to understand how a regression is done. Indeed, I would argue that the easy accessibility of many fine computer programs for performing multiple regressions makes it no longer necessary to teach a student the mechanics of regression analysis. But we are confronted with a vastly increased need to insure that the student is quite familiar with the premises upon which regression analysis is based, and that he understands the various statistical measures which help him to evaluate the results of a regression. In the old days, before the time-sharing computer, such concepts as t-ratios, $\bar{R}$ squareds, and F tests were text book concepts, something to be read about but not used. Now they are a very real part of the everyday work of the student.

Our experience with linear programming is very similar. Once again, a curiosity has become a standard tool, and the student's understanding of the application of this tool requires much more careful thought. One of the difficulties we have encountered stems from the fact that there are few texts available which are really helpful to the student who is equipped with computing power. For the most part, statistics texts and quantitative texts are aimed at explaining the mechanics of a particular type of analysis. This is precisely the element that is least important to the student who has computing power available. Materials have begun to appear in recent years, aimed at the student who uses a computer. But there are still serious deficiencies which an instructor generally must make up for himself.

It is not always obvious to an entering student that the heavy use he sees being made of quantitative techniques does not really involve a grasp of exotic mathematics. The teaching of quantitative techniques in a business school has always presented serious difficulties, and those difficulties can be significantly increased with increasing reliance on computing power. I believe the problem is very largely a psychological one, however. I do not think that it is necessary to teach more mathematics or indeed any more quantitative techniques. It is necessary to teach these techniques in different ways. But there remains a serious problem when the liberal arts undergraduate, with no particular familiarity or liking for quantitative techniques, arrives at a school that so obviously exploits these techniques. I do not think we fully anticipated this problem, and I do not think we have yet discovered just how to solve it.

6. What About "Canned" Programs?

By canned programs I mean computer programs that have been written, perhaps by a teacher or by students, and placed in a library available to all students. There are two kinds of canned programs currently in use at business schools. One kind consists of programs written specifically to solve certain cases or problems used in a course. The students will be given the case or the problem and told to call a certain program out of the system library and use it to arrive at a solution. This kind of canned program has generally been avoided at the Tuck School. Its great disadvantage is that it tends to turn the solution of a case or problem into a routine mechanical exercise. There are a few instances in which this may be unavoidable. By and large, however, our preference has been for canned programs that simply perform standard kinds of analysis, analysis that may be of use in a great many situations but that is not tailored to a particular problem or case. For this reason, virtually all of the programs in the Tuck School program library are programs that any businessman is likely to find useful. They are the kinds of programs you are likely to encounter in the library of any commercial time-sharing service, and indeed many of the programs offered by the commercial services were taken from our library.

Even canned programs of this second kind, however, present some serious problems. I described above the teaching of an understanding of bond calculations by having each student write a program to compute a bond price or a bond yield. I realized when I first used that assignment that it probably could not be repeated in a second year. By that time the Tuck School library included programs to compute bond prices and bond yields. Since all a student has to do is call out one of these library programs and run it, it does not seem to make much sense to ask him to write his own library program to do the same job. In any case, all he has to do is copy the library program. To some extent this problem can be overcome by asking students to devise modifications that can only be made by one who understands how the library program works. For example, the second year, I asked the students to produce a program that would calculate an after-tax bond yield. However, even this variation has its limitations and

I don't believe I can now devise an assignment that will have quite the impact that my first bond calculation assignment had.

Another problem connected with canned programs concerns the use of these programs by a student who is happy to run them without fully understanding how they work, and much more important, the conditions under which they can be expected to give useful answers. I am prepared personally to make some concessions on the first point, in order to concentrate on the second. I know a great many teachers are unwilling to see students using a technique the mechanics of which they do not fully understand. Most teachers for example like to feel that their students are capable of performing a regression analysis by hand before they begin to use a computer to do regressions. I am inclined to think we should begin to change our ways of thinking here. I don't think it is still appropriate to take up student time with the mechanics of analytic techniques when the students will certainly never employ these techniques except through computer programs. I am quite prepared to have students make use of multiple regression analysis, linear programming, quadratic programming, and a number of other quantitative analytic techniques without any knowledge of the mechanics by which the technique is effected. I think we may be well advised to take the time we used to use for teaching the mechanics and spend it on applications, on an understanding of the assumptions lying behind each of these techniques and the interpretation of results. This is a very personal opinion and is certainly not shared by all on the Tuck School faculty.

Almost all of the canned computer programs in the Tuck School library were written by students or faculty at Tuck. Five and a half years ago there simply were no library programs available for time-sharing, at Dartmouth or anywhere else. Subsequently we have built up our own library, and many of our programs have gone through a steady series of improvements and adaptations over the years. For other institutions there is probably no need to repeat this process. Certainly any instructor who makes use of a time-sharing computer is going to develop programs of his own and will want to put them in a library available to his students. But it seems only sensible to make the best use of libraries already in existence.

It has become very clear to me that a crucial aspect of any library program, if it is to be used, is a clearly written manual explaining the use of that program. We have had many programs in our library, some of them excellent programs, that were never used simply because no explanation of the programs existed, and learning to use a program by dissecting it or by trial and error is much too tedious. It is crucial if students are to use a library program to be able to put in their hands a clear manual for the program. I will discuss later on in more detail my experience with manuals.

One remaining problem concerns how to move a program from an existing library to your own computer. Unfortunately, the variety of computer languages and hardware now in use frequently presents serious difficulties in shifting a program from the library at one institution to the library at another. The Tuck School program library, for example, is written in BASIC and resides in the disc storage in a General Electric (now Honeywell) 635 computer. Some of the programs will operate on other computers that make use of the BASIC language. Others will operate only on a General Electric 600 series computer. Converting programs from one language to another and from one type of hardware to another is sometimes possible but always expensive. Whether the ultimate answer lies in more standardization of languages and equipment or in some new sources of financing for the necessary conversions I do not know. The present situation is far from satisfactory.

7. Are We Developing Too Heavy a Reliance by Students On the Computer?

During our first two years experience with time-sharing, as we saw our students making increasingly heavy use of the system and particularly as we saw them dealing with problems that simply could not be handled without a computer, the proposition was put forward that we were teaching our students to be heavily dependent upon a facility that simply would not be available to them after they had left the campus. At that time, however, we were optimistic that computing facilities would become more widely available for problem solving in business corporations and we were also convinced that the heavy use of the computer added so much educational value to the Tuck School program that it was worth taking the risk that students would be disappointed on the job. As it has turned out, for the most part the students who have made the greatest use of computing power and have learned what it can do for them in the solving of business problems, have succeeded in finding employment with companies offering these facilities. A number of our graduates were in fact instrumental in persuading their employers to turn to time-sharing.

Today, vor students intending to work for at least medium sized companies I do not think the danger of reliance on computer facilities poses any serious problems. However, for students who intend to work for quite small companies, companies that are convinced they cannot afford computer facilities for analytical purposes (as opposed to routine data processing) there may be a problem. I am inclined to be as optimistic as I was a few years ago. At that time even the large corporations were not much interested in using computers for solving problems. The cost of compujing is coming down, there are plenty of time-sharing computer services in operation, and I think more small businesses will begin to appreciate the usefulness of computing power.

I am inclined to think however, that some familiarity on the part of the students with the economics of computing may become more important. Our own program has been rather deficient in this respect, as I noted above.

8. A Specific Example: The Investments Course at the Tuck School.

I offer a brief description of the evolution of the Investments course which I teach, as an example of the development of a course simultaneously with the introduction of time-sharing and very heavily influenced by the availability of computing power.

I took over the Investments course at just the time when time-sharing computing first became available at the Tuck School. My first concern was to find or develop some good analytical text material for the students. The text books were disappointing and I therefore began by giving the students journal articles to read. Most of these articles described analytic approaches that involved the use of a computer. Either the analytic technique itself could only be applied by means of a computer (this was especially true of stock selection techniques that involved massaging large banks of data) or a computer was required for the empirical testing of the validity of the analytic approach.

It soon became clear that almost all of the computer applications implicit or explicit in these articles could without great difficulty be programmed on our own computer system. In fact the advantages of the time-sharing system with a simple computer language were so significant that I frequently found that in an afternoon or so I could repeat all of the computer work described in a journal article, work that clearly involved some weeks or months of effort using a conventional batch processing computer system. So I began to assemble a collection of computer programs for applying and testing the analytic approaches described in the journal articles. Some of these programs I wrote for myself. Some were written by individual students, some were written by groups of students as course projects.

As the years went on the programs were refined and elaborated, particularly as the capability of our computer system expanded. And I began to incorporate substantial banks of data into our collection of programs. This made possible the empirical testing of theories and analytic techniques on a fairly large scale. The students were now equipped to read about an interesting analytical approach to investments, to discuss the plausibility of that approach, and to try it out. The trying out frequently led to discoveries that the success of the technique was heavily dependent upon judgements which are very difficult to make, something the author of the technique did not always make clear in his write-up. It sometimes became apparent that there was really nothing to the analytical technique; success or failure was simply a random result. Sometimes it became clear that the technique works or does not work for reasons that were not apparent to its author. In short, the availability of the time-sharing computer made it possible to equip the students to explore fully a variety of theories and techniques that without the computer could only have served as a basis for conjecture.

The final product towards which I have been working in the Investments course, in terms of materials and facilities for students, consists of four parts. First is a text describing analytical theories and techniques that seem to represent the best and most recent thinking. Second is a set of cases and problems, frequently providing opportunities for applying the techniques. Third is a set of computer programs and banks of financial data enabling the student to apply and test the analytic theories and techniques. And fourth is a book of manuals describing in some detail the use of all these programs.

I won't attempt to describe the course in any further detail. The text has been published by Praeger Publishing Company under the title Investments: New Analytic Techniques, and the book of manuals is published at the Amos Tuck School as Manuals for Computer Programs in Finance and Investments. I will add a brief Appendix outlining the computer applications in the Investments course, to try to give some idea of the kinds of programs and data banks which I use.

NOTES

* For a detailed discussion, see Richard Watson, Time-Sharing System Design Concepts, (McGraw-Hill, 1970).

APPENDIX

Investments: Course Topics with Computer Applications

(The computer programs and data banks are all in the Tuck School program library and can be called out on a terminal. All are described in the book of manuals.)

Topic: Measuring Performance of Mutual Funds. Selecting a Fund.

Typical Computer Application:

Analyze the past performance of a group of mutual funds, using various measurements involving risk and profitability. See if there is evidence of consistent superiority of any fund or class of funds. The program library contains a program that will perform many kinds of analysis, and annual data on 218 U.S. and 88 Canadian mutual funds, and monthly data on 70 funds.

Assembling Portfolios of Securities.

Drawing on historical data for a large number of common stocks, use one of three mathematical portfolio selection techniques to assemble portfolios. See if the portfolios do represent a cross section of industries. Test the performance of the chosen portfolios over a subsequent time period to see how well they would have done. Compare the performance with that of portfolios selected at random.

The program library contains data on several hundred common stocks and programs to apply each of three methods of portfolio selection and to evaluate subsequent performances of the portfolios.

Stock Valuation: Determining the Intrinsic Value of a Stock.

Compare the investment value today of a "growth" stock and an "income" stock, using a computer program to establish the intrinsic values of these stocks on the basis of various assumptions. As an aid in making the assumptions, make use of a data file and programs to calculate such features as historic earnings growth, profitability etc.

Filter Techniques: Identifying Stocks to Buy and Sell by Screening Data.

Test suggested techniques for finding under- and over-valued stocks to see whether they would have proved profitable if applied to the stocks for which data are available in the computer.

Income statement and balance sheet data for 10 years for about 100 companies are available in the computer library, as are the programs to screen the data.

Price/Earnings Ratio Analysis to Find Stocks to Buy and Sell.

Compare actual with theoretical price/earnings ratios for a number of stocks, to identify the most over- and undervalued.

A file of 17 items of data likely to be useful in explaining a stock's price/earnings ratio, for 100 stocks for 5 years is available in the library, together with a program to modify any of the data and a program to compute the actual and hypothetical price/earnings ratio.

Bond Switching; Improving the Profitability of a Bond Portfolio.

Given a set of forecasts of trends in interest rates and shifts in the relative popularity of different kinds of bonds, calculate the profitability of selling one particular bond and buying another, and find the conditions under which the switch would prove unprofitable.

Programs to compute bond prices and yields and the profitability of switching (including sensitivity analysis) are in the program library.

COMPUTERS AND COMPUTER-AUGMENTED ACCOUNTING AT LENOIR RHYNE COLLEGE

Craig C. Koontz
Lenoir Rhyne College
Hickory, North Carolina

1. Abstract

Lenoir Rhyne College obtained access to the North Carolina Educational Computer Service in 1968. At present this service, through the Triangle Universities Computation Center (TUCC), is used by sixty schools and colleges. Lenoir Rhyne has an IBM 1050 console and a teletype. The present facilities have already been outgrown, and plans are being made for expansion. Initially the computer was used solely by the mathematics department at Lenoir Rhyne, but now members of the business, physics, and chemistry departments are finding uses for the computer.

Over one hundred students, about one-tenth of the student body, are using the computer terminal for programming courses or for computer-oriented courses. Just this semester, Mathematics 124, Mathematics of Investment, became a computerized course. Over twenty programs, which were written by students at Lenoir Rhyne, are stored on magnetic tape at TUCC and are recalled by students in Mathematics 124 in order to do all the calculations in the course. Planned for the Fall of this year is a computer-augmented course in accounting principles which will be a four-hour course having a two-hour laboratory session each week in the computer terminal.

Other courses being taught at Lenoir Rhyne include Core 110, a course added this year which introduces the student to the concepts of computer utilization and the impact that the computer has on society. The computer facilities are used in Statistics, Mathematics 210, to recall programs that compute statistical information. Numerical analysis, Mathematics 430, is primarily a programming course for mathematics majors. Programs are written to do numerical differentiation and integration procedures. Computer communications, Mathematics 200, is strictly a programming course in which the students learn to write programs in PL/1 (Programming Language/1).

During the Interim period, a course entitled Computer Research was offered. The purpose of this program was to expand the use of the computer at Lenoir Rhyne into many areas where it had not been used previously. Two students took this time to write programs for a business of finance course. Their programs are now being used by the college in teaching Mathematics 124, Mathematics of Investment. I used this time to develop programs which would be of use in teaching a basic accounting course augmented by the computer. These programs are to be discussed in detail later.

Lenoir Rhyne teaches the PL/1 language because it is a most versatile language. It is not strictly a mathematical language or a business language. By teaching a language that every student at Lenoir Rhyne can use, we hope that more students from different disciplines will begin to use the computer facilities.

II. Accounting and the Computer

The Committee on Course and Curricula of the American Accounting Association recommended in 1964 that computer-oriented programs be incorporated into basic accounting courses. Also, they recommended instruction in a basic computer language so that the students would be capable of writing simple computer programs. In general, they urged the incorporation of the computer in undergraduate level courses.

The purpose of the programs included in this paper is to introduce the beginning accounting student to the computer. In doing so, the student will come to realize the capacity of the computer in solving a great variety of accounting problems. These programs were designed to be flexible enough to be used by any student who has had or is taking an introductory accounting course.

These programs have been stored on magnetic tape at TUCC so that they can be recalled easily by the student. In order to obtain the most benefit from the use of the computer, the student should understand the principles behind the calculations before attempting to solve the problem on the computer. There is no intent to alter the basic accounting course. The purpose of the computer is to help eliminate much of the tedious, routine "busy-work" which plagues the accounting student. Thus, the student will be able to spend more time mastering the principles involved rather than using his time to do the tedious calculations.

The student does not need to know the PL-1 language in which these programs were written. He will need only five minutes of instruction in the use of the terminal and the keypunch machine in order to recall a program. The student needs to perform only three simple steps:

1. KEYPUNCH the necessary data.
2. CALL the program by name from the library through the use of job cards.
3. FEED in the data which is to be processed.

On receiving the information, the computer will calculate (where necessary) and print out the desired output. Also, the program will process as many sets of data at once as desired.

The use of library programs (programs stored on discs or tapes) allow the instructor to incorporate a computer-oriented problem into the traditional accounting course. Every business student needs to become aware of the many functions now being performed by the computer with greater speed and accuracy than ever before. One must realize the tremendous potential for expanded computer usage both in the business and educational worlds. A business must realize where its gains (or losses) are occurring. With the aid of the computer, horizontal analysis can be made in several seconds and the printout will show at a glance where these gains (or losses) occurred. Our plans are to correlate computer facilities and the business department with special concentration on the accounting courses.

III.Computer-augmented Accounting Programs

The programs that are to be used in the computer augmented accounting course and their functions are listed below:

1) INCOME--computes and prints out a balance sheet.
2) INCSTAT--computes and prints out an income statement.
3) COMP--computes and prints out a corporation income statement, containing a stockholder's equity section.
4) COMPANAL--computes horizontal comparative analysis of a balance sheet for two years.
5) COMPARI--computes horizontal comparative analysis of an income statement for two years.
6) FIFO--computes the value of ending inventory by the FIRST-IN, FIRST-OUT method.
7) LIFO--computes the value of ending inventory by LAST-IN, FIRST-OUT method.
8) WTAVG--computes the value of ending inventory by the weighted average method.
9) PAYROLL--computes wages, FICA tax, Federal witholding tax, state witholding tax, the hours worked, and the over-time pay.
10) COSTMS--computes a horizontal comparative analysis of schedule of cost of merchandise sold.
11) SHORT--computes a horizontal comparative analysis of a shedule of current assets.
12) RECSE--computes rate earned on common stockholder's equity.
13) EARNPSH--computes earnings per share on common stock.
14) RATEARN--computes rate earned on total assets.
15) SUMOFYR--computes depreciation by sum-of-the-years digits method.
16) STRLINE--computes depreciation by the straight line method.
17) DECBAL--computes depreciation by the declining balance method.
18) UNITPRO--computes depreciation by the units-of-production method.
19) PERPET--computes the value of the inventory perpetually.

IV. Sample Program--INCSTAT

The purpose of this program is to print out an income statement placing the proper total in the proper column, thus saving the student much work. An example of one of the programs, INCSTAT, is printed on the following pages along with the results which can be obtained.

A. Variables Used in this Program:

BEGDATE = The beginning date of the fiscal period.
ENDDATE = The ending date of the fiscal period.
SELEXP(w) = A matrix containing the names of a variable number of selling expenses.
VSELEXP(w) = A matrix containing the values of each selling expense.
GENEXP(x) = A matrix containing the names of a variable number of general expenses.
VGENEXP(x) = A matrix containing the values of each general expense.
COMP = The name of the company.
TYPSTAT = The type of statement (which will be INCOME STATEMENT)
YMODA = 'FOR YEAR ENDED and the date for the end of the next fiscal period.'
NSALES = Net sales
BMI = Beginning merchandise inventory.
PUR = Purchases.
MAVES = Merchandise available for sale.
CMS = Cost of merchandise.
GRPROF = Gross profit on sales.
TSEXP = Total selling expense.
TGEXP = Total general expense.
TOPEXP = Total operating expense.
NETFOP = Net income from operations.
INCTAX = Income tax.
NETINC = Net Income.

B. Program Statements

```
Incstat: Proc options (main);
Dcl(comp,typstat,ymoda) character(35);
Dcl(aa,bb,cc,dd,ee,f,g,i,j,k,l,m,n,o,p,q,r,s,t,u,v) character(35);
Dcl(sales,sra,nsales,bmi,pur,mavfs,emi,cms) fixed decimal(11,2);
Dcl(grprof,tsexp,tgexp,topexp,netfop,difothe,tothinc,tothexp,incbef,
    inctax,netinc) fixed decimal(11,2);
Dcl (e) character(22);     Dcl(h) character(27);
Dcl (begdate,enddate) character(25);
Dcl (selexp(w)) character(40) controlled;
Dcl (vselexp(w)) fixed decimal(11,2) controlled
Dcl (genexp(x)) character(40) controlled;
Dcl (vgenexp(x)) fixed decimal(11,2) controlled;
Dcl (w,x,y,z,a,b,c,d) fixed decimal(2);
Dcl (ff) character(15);
aa='revenue from sales:';   bb='less sales returns and allowances';
cc='net sales';   dd='cost of merchandise sold:';
ee='sales';   ff='_______________';   e='merchandise inventory,';
f='purchases';   g='merchandise available for sale';
h='less merchandise inventory,';   i='cost of merchandise sold';
j='gross profit on sales';   k='operating expenses:';
l='selling expenses:';   n='general expenses:';
m='total selling expenses';   o='total general expenses';
p='total operating expenses';  q='net income from operations';
r='other expense:';   s='other income:';
t='income before income tax';
u='income tax';   v='net income';
Start: get file (sysin) list(a,b);
on endfile(sysin) go to stop;
w=a; x=b;  tsexp=0;   tgexp=0;
allocate selexp(w);  allocate vselexp(w);  allocate genexp(x);
allocate vgenexp(x);
get file (sysin) list (sales,sra,bmi,pur,emi,selexp,vselexp,
  genexp,vgenexp,inctax,comp,typstat,ymoda,begdate,enddate);
put skip file (sysprint) edit (comp) (column(29),a(35));
put skip file (sysprint) edit (typstat) (column(30),a(35));
put skip file (sysprint) edit (ymoda) (column(23),a(35));
put skip file (sysprint);
put skip file (sysprint) edit (aa) (column(1),a(35));
put skip file (sysprint) edit (ee) (column(3),a(35));
put file (sysprint) edit (sales) (column(67),p'$$$,$$$,$$$.$$');
put skip file (sysprint) edit (bb) (column(5),a(35));
put file (sysprint) edit (sra) (column(67),p'zzz,zzz,zzz.zz');
put skip file (sysprint) edit (ff) (column(67),a(15));
put skip file (sysprint) edit (cc) (column(3),a(35));
nsales= sales - sra;
put file (sysprint) edit (nsales) (column(87),p'$$$,$$$,$$$.$$');
put skip file (sysprint) edit (dd) (column(1),a(35));
put skip file (sysprint) edit (e) (column(3),a(22));
put file (sysprint) edit (begdate) (column(26),a(25));
put file (sysprint) edit (bmi) (column(67),p'$$$,$$$,$$$.$$');
put skip file (sysprint) edit (f) (column(3),a(35));
put file (sysprint) edit (pur) (column(67),p'zzz,zzz,zzz.zz');
put skip file (sysprint) edit (ff) (column(67),a(15));
put skip file (sysprint) edit (g) (columnN3),a(35));
```

```
mavfs = bmi + pur;
put file (sysprint) edit (mavfs) (column(67),p'$$$,$$$,$$$.$$');
put skip file (sysprint) edit (h) (column(5),a(27));
put file (sysprint) edit (enddate) (column(33),a(25));
put file (sysprint) edit (emi) (column(67),p'zzz,zzz,zzz.zz');
put skip file (sysprint) edit (ff) (column(67),a(15));
put skip file (sysprint) edit (i) (column(1),a(35));
cms = mavfs - emi;
put file (sysprint) edit (cms) (column(87),p'zzz,zzz,zzz.zz');
put skip file (sysprint) edit (ff) (column(87),a(15));
put skip file (sysprint) edit (j) (column(1),a(35));
grprof = nsales - cms;
put file (sysprint) edit (grprof) (column(87),p'$$$,$$$,$$$.$$');
put skip file (sysprint) edit (k) (column(1),a(35));
put skip file (sysprint) edit (l) (column(3),a(35));
do w = 1 by 1 to a;
tsexp = tsexp + vselexp(w);
put skip file (sysprint) edit (selexp(w)) (column(5),a(40));
put file (sysprint) edit (vselexp(w)) (column(47),p'zzz,zzz,zzz.zz');
end;
put skip file (sysprint) edit (ff) (column(47),a(15));
put skip file (sysprint) edit (m) (column(7),a(35));
put file (sysprint) edit (tsexp) (column(67),p'$$$,$$$,$$$.$$');
put skip file (sysprint) edit (n) (column(3),a(35));
do x = 1 by 1 to b;       tgexp = tgexp + vgenexp(x);
put skip file (sysprint) edit (genexp(x)) (column(5),a(40));
put file (sysprint) edit (vgenexp(x)) (column(47),p'zzz,zzz,zzz.zz');
end;
put skip file (sysprint) edit (ff) (column(47),a(15));
put skip file (sysprint) edit (o) (column(7),a(35));
put file (sysprint) edit (tgexp) (column(67),p'zzz,zzz,zzz.zz');
put skip file (sysprint) edit (ff) (column(67),a(15));
put skip file (sysprint) edit (p) (column(1),a(35));
topexp = tsexp + tgexp;
put file (sysprint) edit (topexp) (column(87),p'zzz,zzz,zzz.zz');
put skip file (sysprint) edit (ff) (column(87),a(15));
netfop = grprof - topexp;
put skip file (sysprint) edit (q) (column(1),a(35));
put file (sysprint) edit (netfop) (column(87),p'$$,$$$,$$$.$$');
put skip file (sysprint);
sixth:  put skip file (sysprint) edit (u) (column(1),a(35));
put file (sysprint) edit (inctax) (column(87),p'zzz,zzz,zzz.zz');
put skip file (sysprint) edit (ff) (column(87),a(15));
put skip file (sysprint) edit (v) (column(1),a(35));
netinc = netfop - inctax;
put file (sysprint) edit (netinc) (column(87),p'$$$,$$$,$$$.$$');
seventh: free selexp;  free vselexp;  free genexp;  free vgenexp;
go to start;
stop:  end incstat;
```

C. Input

1. Sample data

```
5 7 15000000 3150000 910000 10850000 970000 'SALES SALARIES'
ADVERTISING EXPENSE' 'DEPRECIATION EXPENSE-STORE EQUIPMENT'
'MISCELLANEOUS SELLING EXPENSE' 'STORE SUPPLIES EXPENSE'
775000 200000 1920000 25000 20000 'OFFICE SALARIES' 'RENT
EXPENSE' 'HEATING AND LIGHTING EXPENSE' 'PROPERTY TAX
EXPENSE' 'INSURANCE EXPENSE' 'UNCOLLECTABLE ACCOUNTS EXPENSE'
'MISCELLANEOUS GENERAL EXPENSE'  200000 160000 60000 70000
31500 15000 12500 120000'HILL RADIO AND TV' 'INCOME STATEMENT'
'FOR YEAR ENDED AUGUST 31, 1970' 'SEPTEMBER 1, 1969'
'AUGUST 31, 1970'
```

2. Order of Data.

1. The number of SELLING EXPENSES which are to be used.
2. The number of GENERAL EXPENSES which are to be used.
3. The value of SALES.
4. The value of SALES RETURNS AND ALLOWANCES.
5. The value of the BEGINNING MERCHANDISE INVENTORY.
6. The value of the PURCHASES.
7. The value of the ENDING MERCHANDISE INVENTORY.
8. The name(s) of all the SELLING EXPENSES.
9. The value of each SELLING EXPENSE in the same order as the names.
10. The name(s) of all the GENERAL EXPENSES.
11. The value of each GENERAL EXPENSE in the same order as the names.
12. The value of the INCOME TAX.
13. The name of the COMPANY.
14. The name of type of statement.
15. The name for the end of the period.
16. The name of the beginning date of the fiscal period.
17. The name of the ending date of the fiscal period.

Each data item which is indicated in the Order of Data as name(s) must be enclosed in single quotation marks.

D. Printout of Program

```
                           HILL RADIO AND TV
                            INCOME STATEMENT
                      FOR YEAR ENDED AUGUST 31, 1970

 REVENUE FROM SALES:
   SALES                                                         $150,000.00
     LESS SALES RETURNS AND ALLOWANCES                              3,150.00
                                                                 -----------
   NET SALES                                                                 $146,850.00
COST OF MERCHANDISE SOLD:
  MERCHANDISE INVENTORY, SEPTEMBER 1, 1969                         $9,100.00
  PURCHASES
                                                                  108,500.00
                                                                 -----------
  MERCHANDISE AVAILABLE FOR SALE                                 $117,600.00
    LESS MERCHANDISE INVENTORY, AUGUST 31, 1970                     9,700.00
                                                                 -----------
COST OF MERCHANDISE SOLD                                                      107,900.00
GROSS PROFIT ON SALES                                                        -----------
                                                                              $38,950.00

OPERATING EXPENSES:
  SELLING EXPENSES:
    SALES SALARIES                                 7,750.00
    ADVERTISING EXPENSE                            2,000.00
    DEPRECIATION EXPENSE-STORE EQUIPMENT           1,920.00
    MISCELLANEOUS SELLING EXPENSE                    250.00
    STORE SUPPLIES EXPENSE                           200.00
                                              -----------
      TOTAL SELLING EXPENSES                                      $12,120.00
  GENERAL EXPENSES:
    OFFICE SALARIES                                2,000.00
    RENT EXPENSE                                   1,600.00
    HEATING AND LIGHTING EXPENSE                     600.00
    PROPERTY TAX EXPENSE                             700.00
    INSURANCE EXPENSE                                315.00
    UNCOLLECTIBLE ACCOUNTS EXPENSE                   150.00
    MISCELLANEOUS GENERAL EXPENSE                    125.00
                                              -----------
      TOTAL GENERAL EXPENSES                                        5,490.00
                                                                 -----------
TOTAL OPERATING EXPENSES                                                       17,610.00
                                                                             -----------
NET INCOME FROM OPERATIONS                                                    $21,340.00
INCOME TAX                                                                      1,200.00
                                                                             -----------
NET INCOME                                                                    $20,140.00
```

V. Conclusion

The included programs are only a sample of how the computer can be incorporated into the accounting area. There are many areas in which programs can be written that have not yet been investigated. The purpose of presenting these programs is to stimulate others to begin work in the area of computer-augmented accounting. It is in hopes that by examining a simple program, others will realize the tremendous potential for expanded computer usage in the business and educational worlds.

FOOTNOTE

1. American Accounting Association Committee on Courses and Curricula, "Electronic Data Processing in Accounting," The Accounting Review, Vol. XL, No. 2, (April, 1965), p. 442.

REFERENCE

Niswonger, C.R., and Fess, P.E., Accounting Principles, 10th Ed. Cincinnati: South-Western Publishing Co., 1969.

COMPUTER-AUGMENTED ACCOUNTING EDUCATION
AT KNOX COLLEGE

Wilbur F. Pillsbury
Knox College
Galesburg, Illinois 61401

This paper will concern itself with my experiences in offering a one-year course in Computer Augmented Accounting. Most educators are familiar with Computer Assisted Instruction in which the computer is programmed to present information to the student, ask questions, evaluate answers and adapt the instructional process to the student's individual needs. This use of the computer requires a great deal of sophisticated equipment and programs. Thus far, this type of teaching is not within the reach of many colleges and universities. However, computer augmented instruction, I suggest, is available to most inasmuch as its use requires little computer equipment and software. My observations are divided into five parts:

-What Knox College is doing in Computer Augmented Accounting
-Research involved
-Why a computer augmented Principles of Accounting Course
-Results thus far
-Suggestions to consider

WHAT KNOX IS DOING IN COMPUTER AUGMENTED ACCOUNTING EDUCATION

At Knox College the Principles of Accounting course is a one year, three term course offered to students majoring in economics and business administration and to other students within the liberal arts program. The content of the first two terms resembles the typical beginning accounting offered at most colleges and universities. But at Knox College the teaching process has been altered by means of using the computer as a teaching tool.

I have written some 80 computer programs which are stored on the disk of an IBM 1130 computer. These programs can be called by the student to journalize, post, prepare trial balances, financial statements and their accompanying analyses. In addition, programs are available which deal with managerial and cost accounting, actuarial science, inventory analysis, capital budgeting, and the accounting decision making process.

The student uses his creative thinking about accounting to prepare data for key punched cards. The computer then takes over for the time consuming and tedious computation which usually involves no learning whatsoever. During the first two academic terms the computer is used as a tool to more effectively teach accounting. Discounting the saving in time and drudgery, which by the way is tremendous, the student, without understanding computer programming, already can comprehend the powerful uses of this new tool at his command.

In these first two terms, the students are given typical accounting problems for which they key punch the raw data on cards. If the student is weak in a particular phase of the course, he is urged to do other problems, make up his own data, indeed he can create his own complete practice set, using the variables which have been programmed and stored on the computer disk. One might state that the possibilities for learning are limited only by the student's interest, time, and imagination.

During the third term, the positions are reversed; that is, the accounting process and the problems are used as a means of teaching computer programming. It is in this term that the student learns how to use the computer by means of a user oriented language. During this term the students are taught to write, in the FORTRAN language, the very programs which they have been using the first two terms.

In summary, then, during the first two terms the students use the computer as a tool to learn accounting. During the third term the students use accounting as a tool to learn computer programming. The learning process is reinforced by this complementary interaction of the computer and the principles of accounting.

During these three terms all of the recommendations regarding the computer and the accounting student set forth by the American Accounting Association and the American Institute of Certified Public Accountants are fulfilled. Both of these organizations, in strongly worded statements, have insisted that the accounting and business student today must be involved with the computer.

The computer programs which we use are intended as an augmenting device. They are not designed to replace conventional courses or textbooks but to supplement and blend in with them.

Accompanying the computer programs are two workbooks, called "CompuGuides", which can be used in conjunction with any one of the top principles of accounting texts in use today (1). The CompuGuides require only minor modification in the order of presentation for some texts.

To provide a rough idea of the procedure used it might be helpful to illustrate what can be done on a small computer like IBM 1130. Listed below is a partial Table of Contents of CompuGuide One and Compuguide Two. Each chapter represents one or more computer programs which can be called by the accounting student and used to solve an accounting problem given in the CompuGuides or in any of the typical accounting textbooks.

COMPUGUIDE ONE

1. Balance sheet program using only three accounts: Assets, liabilities and capital
2. Balance sheet program using several accounts
3. The recording procedure and the chart of accounts
4. Adjusting entries for apportionments
5. Adjusting entries for accruals
6. Closing entries
7. Accounting for merchandise operations
8. Comprehensive exercise: Tom McCann Shoe Company
9. Short-term financing and simple interest
10. Investment and interest accounting: Compound interest
11. Valuation of inventory using the lower of cost or market
12. Gross profit method of inventory estimation
13. Depreciation of plant assets
14. Bank reconciliation
15. Computerized payroll system

COMPUGUIDE TWO

1. Estimation of goodwill
2. Pricing of bonds
3. Contract and effective rates of interest on bonds
4. Sinking fund
5. Amortization of premium or discount on bond investment
6. Process cost systems
7. Specific order cost accounting
8. Variance from standard cost accounting
9. Present value
10. Present value of an annuity
11. Differential analysis: Sell or lease
12. Differential analysis: Replacement of equipment
13. Discounted cash flow
14. Break-even analysis
15. Price level accounting
16. Financial analysis: 4 key ratios
17. Financial analysis: 13 key rations

RESEARCH INVOLVED

For a man over 30, I had best begin this section with a personal confession. Three years ago last September my knowledge of computers was nearly zero. Knox College had just installed an IBM 1130 computer to be used for educational purposes. I am one of those faculty who has been forced, urged, shown the wisdom of, or haggled into using the computer. I would have been made to feel incompetent if I did not. As it was, I just plain "caught the bug." A few of my students wanted to start using the computer to solve some of the problems dealing with financial statement analysis, so it became evident that I had to learn about the computer in order to keep up with them.

So, I took, along with many students, the non-credit course in computer science which was offered to students and faculty. I made mistakes. I sat there wide-eyed and in wonderment while the students were nodding their heads that they understood what was going on. The students were teaching me--and in some cases they still are. But all of this experience whetted my appetite to see what could be done to bring the computer into our teaching procedure. I took a two-term sabbatical, beginning with one month at an IBM education center. Following that month at IBM, I spent two months at Stanford University and then visited 10 different schools in the State of California to study their application of the computer to their educational process.

WHY COMPUTER AUGMENTED ACCOUNTING

The underlying philosophy of using the computer is to facilitate the learning of the basic concepts and techniques of accounting and to increase the impact of the learning experience. Educational psychologists believe that learning is most effective when three elements are present:

-the learner is actively involved in the process
- repetition is present
-reinforcement and results are provided

Using the computer as a teaching tool is designed to take advantage of these learning principles.

Once this philosophy had been thought through, I struggled with the questions: Do I teach computer programming first and the accounting course afterwards? Or do I teach the course material with the aid of the computer and computer programming after that?

I settled on the latter method believing that there is more logic to this approach. One need only look at the current methods of teaching a foreign language. Students learn to speak the language first via language laboratories; then grammar and sentence structure come more easily. The use first, details later approach is efficient and faster. Or, look at the accounting discipline where the so-called "principles" chapter comes only after the students learn the procedures, rules and methods.

Furthermore, it was possible that if the students learned programming first they would be thinking of programming instead of the accounting course content. Lastly, I did not believe there should be a pre-requisite for the first year accounting course. Therefore, I selected the method whereby the students learn the accounting course material with the aid of the computer and then program writing the third term.

The underlying philosophy of using the computer includes four primary goals:

1. To allow the accounting student to become an intelligent user of available computer facilities.

   At the present time many students (and some professors) are over-awed by the computer. Students who use the computer in their education begin to appreciate how relatively simple the computer is and begin to appreciate the potential of this new tool in accounting and general business usage. Many students discover that computer programming, heretofore a mystery, is relatively easy and definitely fascinating.

2. To use the computer as a tool to better understanding the accounting material being studied.

   The basic logic, reasoning and theory of accounting are still present in the use of the computer in teaching. The student must learn the fundamentals of accounting; the computer in no way performs this thinking and reasoning for him.

   Using the computer as a tool to learn accounting enables the student to do more creative work. Handwritten problems persist at certain stages of the course where the computer cannot be used effectively. However, most accounting concepts and problems can be worked out far more effectively on the computer.

   Each section of CompuGuide material has prepared problems with a known answer.

These problems are not unlike those found in the typical accounting text. However, to aid learning, each section also contains an exercise in which the student applies self-created data and information. Thus, he is able to test his knowledge with his own data.

3. To save the time of faculty and students.

   By means of the programs stored on the machine's disk, the computer performs most of the tasks of organizing, calculating, and record keeping. The time-consuming, uncreative tasks of posting, balancing, preparing trial balances, dividing for ratios, and paper shuffling are all performed by the computer. This saving of time can be spent on developing the arts of analysis, simulation, interpretation and decision making.

   The emphasis on theory, logic, and principles allows accounting to be an interesting and stimulating discipline rather than the dull, clerical routine beginners often find it to be. Key punching time is kept to a minimum by means of a free-form subroutine whereby the student can enter his data in any column of a card providing he leaves more than one space between sets of numbers. Thus, the emphasis is placed upon accounting concepts--not keypunching.

4. To extend some quantitative analysis in the basic accounting course.

   Accounting has been taught for the most part as a science of classification. There has long been a need to develop analytical and management accounting using a more sophisticated level of mathematics.

   With the use of the computer programs both the mathematically and non-mathematically oriented students are able to apply mathematics to accounting. Complex problems, once believed insoluble by the beginning accounting student, may be not only solved, but analyzed and discussed.

RESULTS THUS FAR

Starting out in the first year with the grandiose idea of making a truly scientific test of my experiment, I planned to have three laboratories of students chosen at random in the Registrar's office:

- -One-third of the students were to do 90% of their problems by hand and 10% by computer.
- -Another third of the students were to do 50% of their problems by hand and 50% by computer.
- -A final third of the students were to do 10% of their problems by hand and 90% by computer.

With this procedure, and subsequent testing, I proposed to have a more scientific indication of the effect of using the computer. This scientific experimentation soon collapsed. After about one week the students randomly selected to do 90% and 50% of their work by hand complained bitterly of their extra drudgery and time. "Why do we have to do this extra work?", they objected. "When are we going to learn more about what the computer can do for us?", they asked. "This is all in the interests of a scientific experiment," I told them. This explanation proved insufficient and soon my experiment collapsed under the weight of student pressure.

At the present time about 70% of the problems are done on the computer.

Therefore, it is impossible to give you scientific proof of the worthwhileness of this method of teaching but I have some subjective judgments with a few interesting sidelights:

1. I believe I am teaching 25% more accounting material in 25% less time. I am covering much quantitative and analytical material never covered before. Aspects of "managerial accounting" never discussed in previous courses are now being covered.

2. Students are attracted from a variety of major fields--English, art, music, philosophy, biology, education, chemistry, physics, in addition to the usual economics, mathematics, and political science areas.

3. Only two students in the last two years have failed the first or second term course. As far as I can remember this is a record in some 24 years of teaching. I believe my standards are higher now than at any time in the past. I constantly found myself telling the weak student to use the variables in the programs which were stored on the disk, make up his own problems, and experiment with various phases of the problem.

   In this manner, the computer library programs have all the educational benefits of a "live" laboratory demonstration. The student is allowed to think of, and through, a transaction himself--all with a minimum of supervision. The learning process is thereby reinforced.

4. Two years ago 52 of the 53 students returned for the second term of accounting. Last year 63 of 65 students taking the course returned. These figures are records, too, I believe. All of the science and humanities students returned. It is interesting, two of the three students "lost" were economics majors who switched to computer science.

5. Using the computer approach had forced me to completely change my teaching procedure. This effect may be the most important one. If most faculty are like me, they often enter the classroom with "canned" problems or questions and then start off the discussion by asking, "Well, did you think of this angle?" Usually the students have not and we can assert our superior intellect by putting a different phase of the problem on the blackboard.

   It was a considerable shock to me that this procedure could no longer be used. I have found that as the students were forced to think through their own problems, they, not I, had thought through all of the angles of a problem. The philosophy, art, and music students were asking me, "Well, did you think of it this way?" Frequently I had not.

SUGGESTIONS TO CONSIDER

It is not my idea to make sophisticated programmers out of the students in this course. I do attempt to make them intelligent users of the computer knowing when it should be used and when it should not. I also hope to make them understand that there are both advantages and disadvantages to the computer's use.

Users they are becoming. I feel like I am hanging onto their coattails as they are running out ahead of me. It is not at all unusual now to come down to a class at 8:00 A.M. and see the students come from the computer workroom bleary-eyed from staying up all night in order to prove the problem, or the computer, or me wrong.

So, my first suggestion is to urge you not to get into this type of teaching unless you are willing to work as probably you have never worked before. Your students are going to force you to make changes and really put to a test some of your "sacred cows" of teaching.

These are not times on a college campus when the students are going to put up with the professors with "yellowed-with-age" class notes. Our students are demanding more. I believe the use of the computer as a teaching tool is a partial answer to that demand.

Another suggestion is that I believe we have to start gearing our courses for the coming students who will have used computers in high schools. These students simply are not going to put up with the deadly work of hand written problems or practice sets or pounding the calculator by the hour for ratios for financial analysis.

These students who have been exposed to the computer before they come to us are going to insist that the deadly paper-shuffling and laborious tasks be handled electronically. There will then be time for the analysis, interpretation, and simulation and for truly creative learning to take place.

A third suggestion involves the computer center directors, or the departmental chairmen with faculty members whose class notes are "yellow-with-age". Urge these department heads to take some of their brighter majors and ask them to do some of their problems or current research on the computer. Ask them to take it to these professors with old class notes to inquire if they can do some of their assignments on the computer. In other words, tease the faculty into "catching the bug".

A fourth suggestion is to remember that it is often the younger faculty members who will most likely appreciate this type of teaching. It is they who have had the use of the computer in their graduate work and can more readily appreciate what it can do.

It is a little disconcerting to an old professor like me with some 24 years of standing in a classroom talking--to suddenly discover that, in many ways and in many kinds of material, the students can learn more and understand better if they are left to themselves. My experience with using the computer in teaching is that once the student is presented the ideas and concepts, the use and application of the theories can better be covered by the student working alone or in small groups. In other words, they don't need me!

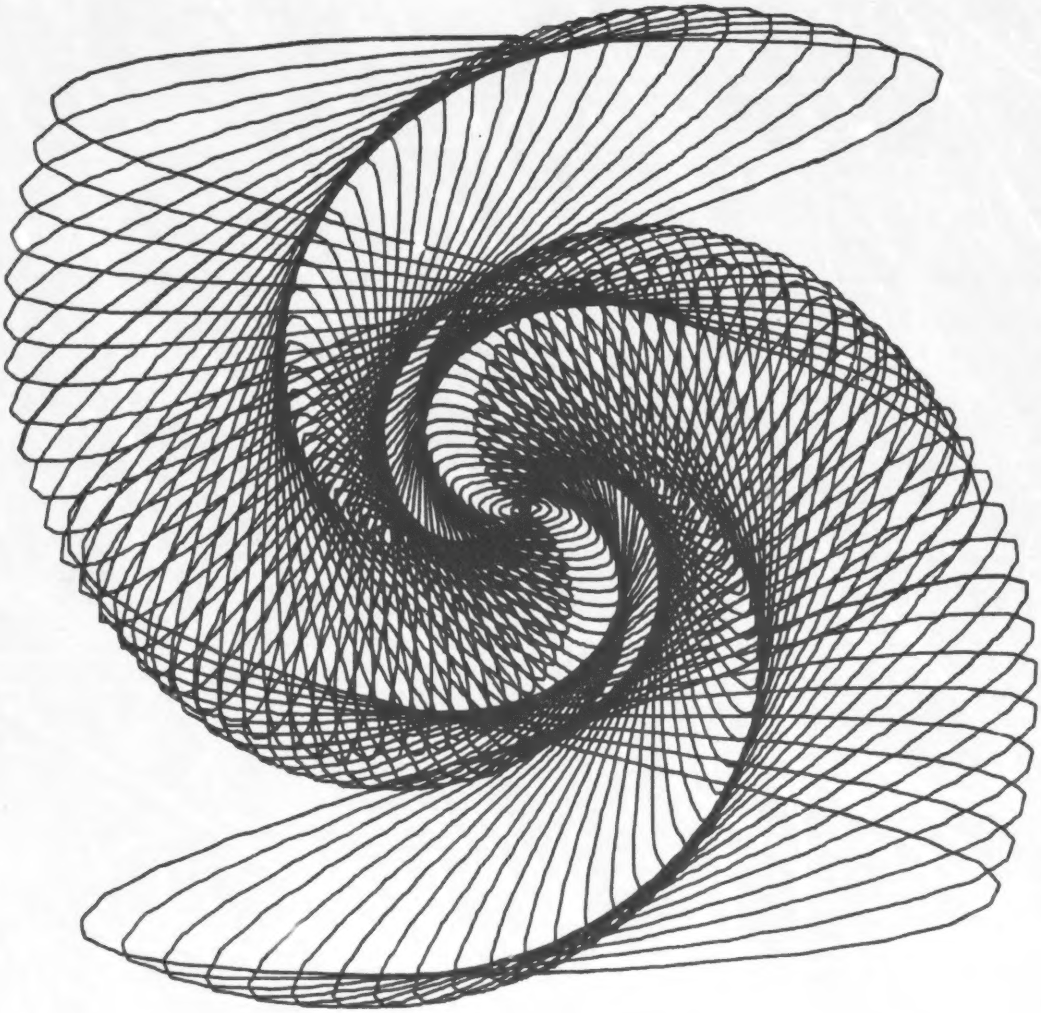
It is in this aspect the older faculty will resist. Our old theories about having to stand up and talk, talk, talk get in students' way. With adequate preparation and given free rein I have found that I am out there trying to catch up with them. With the use of the computer organization and availability are the main functions of the teacher, not lecturing.

In conclusion, someone once said that man is still the lowest cost, 160 pound, nonlinear, all-purpose computing system that is capable of being mass produced by unskilled labor. To this I would add but one bit of counsel: It takes time to work into this type of teaching. Adjustments have to be made--in teaching assignments, in the computer center personnel, in the equipment an institution has available. Therefore, I urge you to go slowly with using a few faculty and only some sections of classes. You are doing a disservice to your institution, to your students, and to yourself, if you encourage immediately that all faculty and all students try to do all of their problems on the computer.

A slower, trial approach will assure satisfied faculty and, I promise you, enthusiastic and enlightened students.

FOOTNOTE

1. Computer Augmented Accounting, CompuGuide One and CompuGuide Two, by Wilbur F. Pillsbury, (Cincinnati: South-Western Publishing Company, 1970 and 1971).



Stephen Toll

OTHER SUBMITTED PAPERS

BIOLOGY

BATES W K (The University of North Carolina at Greensboro)
"Introduction to Computer Methods in an Undergraduate Biology Curriculum"

GASSER W and GEHRT K M (Quincy College, Quincy, Illinois)
"A Computer Program for Identifying Microorganisms"

HOWLAND H C (Cornell University, Ithaca, New York)
"Algebraic Language Instruction in the Biological Sciences Curriculum at Cornell"

HUFFMAN D M and HUTCHINSON F E (Central College, Pella, Iowa)
"Computer Programs as Instructional Aids for Genetic Coding, Genetic Transcription, and Mapping of Genetic Information"

PROCACCINI D J (Emmanuel College, Boston, Mass.) and MARKS L S (St. Joseph's College, Philadelphia, Pa.)
"Biological Complexity and Computer Biology, Some Applications and Ramifications"

SMITH E (Westminster College, New Wilmington, Pa.)
"Use of APL by Non-Science Majors in a Required Science"

SPAIN J O (Michigan Technological University, Houghton, Mich.)
"Simulation in Biology Using an (Olivetti) Desk Top Computer"

BUSINESS ADMINISTRATION

CARSON J E and SWARTZ G B (Monmouth College, West Long Branch, N. J.)
"Use of Computers in Teaching Statistics to Undergraduate Business Administration Students"

HILLMAN A D (Drake University, Des Moines, Iowa)
"Enhancement of the Undergraduate Business Student's Computer Experience--A Plan"

HOLMES J D J (St. Andrews Presbyterian College, Laurinburg, N. C.)
"New Approaches to Business Simulation"

HUNREKER W A (New Hampshire College, Manchester, N. H.)
"Integration of an MIS Curriculum in a Business College"

LOSCHETTER R F and PILAND W E (Moraine Valley Community College, Palos Hills, Illinois)
"A Computer in Every Classroom: How to Get the Computer in the Classroom"

WINTER S J (College of Santa Fe)
"Some Experiences in the Use of a Desk-Top Computer in Teaching Financial Statement Analysis"

CHEMISTRY

BERKA L H, CHRISTIAN P A, GARAWAY M W (Worcester Polytechnic Institute, Worcester, Mass.)
"Laboratory Diagnostic Programs"

DEAL R M (Kalamazoo College, Kalamazoo, Michigan)
"Uses of Computers in Compensatory Science Education in Higher Education"

KIM H (University of Missouri, Columbia, Missouri)
"Applications of Santa Barbara Teleputer Terminal in Undergraduate Physical Chemistry Laboratory Course"

RICHARDS R R (Greenville College, Greenville, Illinois)
"The Programmable Calculator as an Instructional Aid"

SHERMAN L R (North Carolina A & T State University, Greensboro, N. C.)
"Application of 'Canned' Programs and Computerized Literature Searching To An Undergraduate Chemical Curriculum"

COMPUTER SCIENCE

BERNSTEIN N (Northeastern University, Boston, Mass.)
"Computer Simulation as a Technique in the Teaching of Computer Structure"

DIEHR G and PRATER G (University of Washington)
"Batch BASIC--A Low Cost Introduction to Programming"

MINSHALL J A (Coe College)
"Multidisciplinary Applications of a Data Packing System for Small Computers"

POLENZ R C (Wisconsin State University - Eau Claire)
"The Computer Utility as a Small University Resource"

VINCENZI J (Saddleback College, Mission Viejo, California)
"A Computational Devices Course-Or-Learning on What You've Got"

ECONOMICS

MUKHERJEE T (Appalachian State University, Boone, N.C.) and LAUGHLIN W B JR (Fairmont State College, Fairmont, West Virginia)
"Computer-Related Curriculum Development in Economics"

OSMAN J W (San Francisco State College, San Francisco, Cal.)
"Computer Applications in Undergraduate Economics"

SIEGAL P H (Bloomsburg State College, Bloomsburg, Pa.)
'The Use of the Computer in the Undergraduate Curriculum in Business Administration and Economics"

EDUCATION

KULLER R G (Northern Illinois University)
"An Introduction to Algorithms and Computing for the Liberal Arts Student"

ANDERSON E L (Oregon State University)
"The Computer in Teacher Preparation--A Beginning"

MANDELARE J J (Oregon College of Education, Monmouth, Oregon)
"A Computer Applications Course for Teacher Education"

PORTER C M (Oregon State University, Corvallis, Oregon)
"Some Suggestions for Establishing Guidelines in the Development and Implementation of Computer Materials in Undergraduate Instruction"

ENGINEERING

CHIANG D C (Rose-Hulman Institute of Technology)
"A Short History of Analog Computer Usage in Mechanical and Aerospace Engineering Department at Rose-Hulman Institute of Technology"

GUFFNER G E (Buffalo State College)
"The Computerized Take-Home Exam"

LEEPER R (Indiana Institute of Technology)
"Simulating Logic Gates in FORTRAN"

LEVITT J H (Pratt Institute, Brooklyn, N. Y.)
"A Computer Approach to Teaching Small-Signal Analysis"

MUELLER G V (Clark County Technical Institute, Springfield, Ohio)
"Use of a Computer in Teaching Electric Circuit Principles"

NUNN W M JR (Florida Institute of Technology, Melbourne, Florida)
"The Use of Large-Scale Digital Computers in a Modern Undergraduate Microwave Laboratory"

PASKUSZ G F (University of Houston, Houston, Texas)
"Undergraduate Instruction in Computer Use: A Computer-Aided Design Course"

SIEGEL C M (University of Virginia)
"Computer Aided Practice in Electrical Engineering Education"

HUMANITIES

MILLER L W (Montgomery College, Takoma Park, Maryland)
"A CAI Project in English Composition--Experiences, Difficulties, Unsought Side Effects"

URION H K and MANASSE F K (Dartmouth College, Hanover, N. H.)
"Artificial Intelligence: The Key to Computer Assisted Instruction"

WALSH J (University of North Carolina, Chapel Hill, N. C.)
"The Computer as a Teaching Device: The Prospects of Technological Aid for the Humanities"

MATHEMATICS

BITTER G G (Arizona State University) and MAULAND L E (University of Denver)
"Computer Extended Calculus and its Evaluation"

BRASCH B E (University of Denver, Denver, Colorado)
"Computer Extended Instruction in a Remedial Pre-Calculus Course"

EISENBERG S M (University of Hartford, West Hartford, Conn.)
"Computer Oriented Calculus"

GROSS T A (East Carolina University)
"Some Experiences in Using the Computer as an Aid in Teaching College Algebra"

LESER W H (Franklin and Marshall College, Lancaster, Pa.)
"Using a Computer in the Linear Algebra Course"

MCALLISTER M N (Moravian College, Bethlehem, Pa.)
"Some Do's and Don't's in Computer Calculus"

MCNAMARA J (Manhattan College, Bronx, N. Y.)
"NO-NOR Method for Roundoff Errors"

MAY L R (Kennesaw Junior College, Marietta, Georgia)
"Use of the Computer in a Traditional Calculus Course"

MENDELSON B (Smith College)
"Experiences with a Computing Supplement to Freshman Calculus at Smith College"

MINI-COMPUTER

FINEGOLD L (University of Colorado, Boulder, Colorado)
"A Mini-Computer in a Senior Physics Laboratory"

LAFRANCE J E and EBELING C (Wheaton College, Norton, Mass.)
"An Approach to Introductory Computer Science Using a Time-Shared Small Computer Simulation"

LINDAUER M W (Valdosta State College, Valdosta, Georgia)
"Some Applications of Data Acquisition Systems in the Undergraduate Science Curriculum"

STICHA P E and MINNIER C E (Essex Community College, Baltimore County, Maryland)
"Introducing the Instructional Computer at a Small Two-Year College"

PHYSICS

BARRETT R P (Messiah College, Grantham, Pa.)
"From Experiment to Mathematical Model via Computer"

CHOW P (San Fernando Valley State College, Northridge, California)
"Computer Solutions to Schrodinger Equation"

JACOB R L (Cornell College, Mt. Vernon, Iowa)
"Computer Graded Homework in the Introductory Physics Course"

LUNETTA V N (University of Connecticut, Storrs, Conn.)
"The Design and Evaluation of a Series of Computer Simulation Dialogs in Introductory Physics"

LOVE N D (Maryville College)
"Use of the Computer in Maryville College's Beginning Physics"

MUIR R B and CLARK C B (University of North Carolina at Greensboro)
"The Use of Interactive Computing in the Introductory Physics Laboratory"

SEKI R (San Fernando Valley State College)
"An Example of Computer Application in Sophomore Physics Laboratory"

SPANGLER J D (Kansas State University, Manhattan, Kansas)
"An Interterm Undergraduate Symposium on Computational Physics"

WU C (U.S. Naval Academy, Annapolis, Md.)
"Computer Aided Instruction in Thermodynamics at U.S. Naval Academy"

PSYCHOLOGY

CARTWRIGHT G P and CARTWRIGHT C A (Pennsylvania State University)
"A Computer-Assisted Instruction Course in the Early Identification of Handicapped Children"

DENK J R (N. C. Educational Computing Service)
"Homan's Model of Human Interaction--A Case Study of a Model Turning-on a Computer Network to Sociology Applications"

PEPYNE E W (University of Hartford)
"An Integrated Model for Counseling Research, Education and Evaluation"

STARTSMAN T S (Wake Forest University, Winston-Salem, N.C.)
"A Computer Approach to the Generation and Analysis of Psychological Data as an Adjunct to Instruction in Quantitative Methodology"

SOCIAL SCIENCE

ALEXANDER R and JOHNSON C (Texas Southern University)
"A Computer Simulation of Exogamous Marriage Rules in Rural Black Communities"

BUCKLEY D W and D'AMATO C R (U.S. Naval Academy, Annapolis, Md.)
"Simulations in Undergraduate Education: An Initial Experience"

CAMFIELD M A (East Tennessee State University, Johnson City, Tenn.)
"Computer Use in Social Service Research"

DODGE D (Macalester College)
"A Computer Simulation Model for Prediction of Voting in the United Nations"

FELTON W S JR (Keene State College, Keene, N. H.)
"The Use of the Computer in an Undergraduate Seminar in Survey Research"

HUANG J C and MORRISON K C (Tougaloo College, Tougaloo, Mississippi)
"The Transfer of Political Control in a Biracial Mississippi Town: The Measurement of a Black Perspective"

JACKSON G C (Bloomsburg State College, Bloomsburg, Pa.)
"Trying to Combine FORTRAN Procedures with Social Science Substance"

MURACO W A (Wright State University, Dayton, Ohio)
"Computer Assisted Instruction in Introductory and Advanced Undergraduate Geography Courses"

SCRIPTER M W (San Fernando Valley State College, Northridge, California)
"Using the Digital Computer to Teach Quantitative Methods to Undergraduate Geography Students"

SEGELBAUM R S (N.Y. University), WALLS S J (Barnard College), ETZKOWITZ H (Washington Univ.)
"An Experiment in the Use of the Computer as a Medium for the Development of Problem-Solving Ability in Disadvantaged Children"

SHERBENOU E L (Northern Illinois University)
"Teaching Political Science with Computer Assistance"

TAYLOR C L (Virginia Polytechnic Institute) and TANTER R (University of Michigan)
"Rationale for and Progress Toward a Computer Aided International Relations Teaching Package

TAYLOR R E (University of California, Riverside)
"The Utilization of a Computer-Assisted Learning System (CAL) in Undergraduate Teaching: Its Use in Introductory Anthropology Courses"

WHITHED M H (Temple University, Philadelphia, Pa.)
"The Computer, Political Simulation and the Educational Process"

STATISTICS

BOLLAND T W (University of New Hampshire, Durham, N. H.)
"Simulated Statistical Decision-Making: A Method to Gain Classical Sophistication via Bayesian Methodology and a Computer"

BROUSSEAU A (Centre College of Kentucky, Danville, Ky.)
"Integration of Programmable Calculators into the Elementary Statistics Course"

GRUNLOH M (Quincy College, Quincy, Illinois)
"Computer Use in an Interdisciplinary Basic Statistics Course"

KAPLON H (Towson State College, Towson, Maryland)
"Motivation and Meaning for Hypothesis Testing"

KOH Y O (University of Nevada)
"Progesss Report on the Sierra Method (The Tutorial System for Statistics with Time-Sharing Computers"

SHERR L A (University of Kansas)
"The Use of Real Time Computing in Teaching the Basic Course in Probability Theory and Statistics"

SMITH W A (The University of Tulsa, Tulsa, Oklahoma)
"The Computer as a Humanizing Catalyst in Descriptive Statistics"

TANIS E, NEWELL J, DIGGELMANN H (Hope College, Holland, Michigan)
"The Central Limit Theorem and Mixed Distributions"

UNDERGRADUATE INSTRUCTION

ANDERSON G C (Hamline University, St. Paul, Minn.)
"The Small College: Interactive Time-Sharing?"

ARCHIBALD J A JR (State University of New York, Plattsburgh, N. Y.)
"On the Use of Individual Projects in an Interdisciplinary Approach to Computer Science"

BANATHY B A (Monterey Peninsula College, Monterey, California)
"Increasing the Educational Utility of the IBM 1620"

BREITLER A L (Montgomery Community College)
"Introducing the Computer at a Community College"

BROWN W A (Western Washington State College, Bellingham, Wash.)
"A Computer Examination Compositor"

CARLSON S B (Whitman College, Walla Walla, Washington)
"Computer Assisted Research in the Undergraduate Curricula"

CHU Y (Skidmore College)
"Computer-Analyzed Student Responses as a Basis for Improving Undergraduate Instruction in Eleven Academic Disciplines in a Liberal Arts College"

GOLSON J P (Oglethorpe University)
"Student-Financed Computing"

MALKIN J G (University of Texas, Austin) and MCKAY L R (St. Stephen's Episcopal School, Austin)
"Introducing the Computer into Small College and Secondary School Curriculum"

SCHOFIELD J (Bates College)
"The Computer in the Undergraduate Curriculum: General Report from a Small College"

SCHULTZ G W and LEEPER H H (St. Petersburg Junior College, Clearwater, Fla.)
"Calculus Computer Project at St. Petersburg Junior College--BASIC Language"

STERLING T D and POLLACK S V (Washington University, St. Louis, Missouri)
"Implementing a 'Universal' Introductory Course in Computer Science"

VITELLI J L (University of Minnesota)
"Computer and Television: A Joint Venture for Education"

WILLIS P M (Hood College)
"Progress in Computer Science at Hood College"

MISCELLANEOUS

DOUGLAS H L and LOTT T (Burlington County College, Pemberton, N. J.)
"Computer Managed Instruction as an Aid to Instructional Validation"

MUIR R B and HERMAN H B (University of North Carolina at Greensboro)
"MAGIC--A Program for the Management and Analysis of Grades in an Individual Course"

REINFELDS J and JEDLICKA H (University of Georgia, Athens, Georgia)
"Some Applications of AMTRAN"

BEJNAR W (New Mexico Highlands University, Las Vegas, N. M.)
"Continuous Progress Program in Survey of Earth Science"

DENK J (N. C. Educational Computing Service)
"Exchange of Applications Programs for Education--A National Stalemate"

AUTHOR INDEX

Ronald E. Anderson 135, 263
G.F. Atkinson 456
William M. Baird 75
Marion J. Ball 187
Theodore J. Benac 18
Carol Bennett 463
Paul Benson 379
David V. Blagg 371
Ronald Blum 51
Alfred Bork 4
Stephen H. Bryant 171
Victor Bunderson 10
Francis Campos 429
Ronald W. Collins 99
Stephen J. Cutler 371
James A. Davis 382
Craig Decker 253
Thomas R. Dehner 117
Joseph R. Denk 347
David P. Doane 309
Thomas J. Dudley 315
J.F. Eagleston 277
Robert Ehrlich 64
Samuel A. Elder 299
D.M. Ferguson 323
J. Ronald Frazer 322
Robert W. Gill 171
Sabin Head 424
Fred E. Heidrich 214
P. Hlavaty 489
Frederick M. Hornack 359
Floyd R. Horowitz 50
Paul L. Houston 363
Richard Johnson 419
K. Jeffrey Johnson 130
John H. Karl 289
Tim G. Kelley 195
Gene A. Kemper 27
R.N. Killam 46
Charles E. Klopfenstein 269
Craig C. Koontz 508
Glen D. Kreider 142
Thomas E. Kurtz 267
Lois Langland 266
L. Carl Leinbach 13
James Livingston 429
Elijah P. Lovejoy 429
Arthur W. Luehrmann 1
F.E. Lytle 277
Dana B. Main 424
J.C. Mathews 23
Terry F. Mellon 171
David M. Messick 429
Edmund D. Meyers 159
David Milne 379
Joan Mitchell 463
Andrew R. Molnar 251
Mark Monroe 469
Charles A. Moore 411
David Morris 398
John A. Naylor 75
Bruce E. Norcross 117
D.H. Offner 389
H. Malcolm Owen 176
Herbert D. Peckham 68
S.P. Perone 277
James F. Peters 238
Wilbur F. Pillsbury 516
Stephen Prager 339
Paul N. Rappoport 82
Laurence S. Rosen 142
Darlene Russ 94
Arthur Sandusky 429
Richard N. Schmidt 437
Ben R. Schneider 41
Jolayne Service 444
Harvey A. Shapiro 309
Jo-Anne Sharac 94
Bruce Sherwood 463
Donald L. Shirer 58
Francis M. Sim 142
Paul A. Smith 475
J. Laurie Snell 221
David L. Sparks 417
Edward L. Spitznagel, Jr. 217
Jeffrey I. Steinfeld 363
Philip E. Stevenson 112
David G. Sutton 363
J.C. Tan 181
Charlene Tenczar 463
Bruce R. Thomas 211
J. Richard VandeVelde 230
Brian C. Vargus 365
Claude T. Ware 315
George Weber 379
Fred W. Weingarten 266
Douglas White 365
John K. Wildgen 152
Charles L. Wilkens 269
J. Peter Williamson 499
Kent R. Wilson 214
Kenneth C. Young 309